# TV Holography

## 1. Aims:

- To introduce interferometric techniques capable of measuring surface deformations at the sub-micron level.

## 2. Objectives:

1. To understand the origins of speckle patterns and to construct an interferometer to measure surface deformations of an object.
2. To use ImageJ software to manipulate digital images to form correlation fringes.
3. To investigate the voltage response of a Piezoelectric Transducer using TV Holography techniques.
4. To use acquired skills to investigate additional surfaces of the student's choosing.

## 3. List of Apparatus:

- HeNe laser and mount (632.8 nm)
- Lenses and mounts
- The aluminium screen on rotation mount with PZT actuator
- High voltage amplifier (for driving the PZT)
- Beamsplitter and mount
- Aluminium screen in mount for reference path
- Canon EOS 1100D digital camera
- A computer for collecting images and running ImageJ
- An optical breadboard

## 4. Risk Assessment:

### 4.1 Self-risk Assessment:

| **Hazards:** | **How to prevent risks:** | **Risk level:** |
|---|---|---|
| Food and drinks, risk of slipping and shorting electric circuits | - Don't bring food or drinks into the lab | LOW |
| Laser in the eye (using 1mW, 652.8 nm laser) | - Keep above eye level, keep the laser light within the assigned dark box, at a low power level so (rule of thumb) will not damage the retina unless blink reactions are affected (i.e. no drinky drink) | LOW |
| Operating in the dark, risk of tripping | - Keep bags and other loose items away from the operating area | LOW |
| RSI from computer use, eyestrain from work in darkness | - Take regular breaks when using a computer and move around | LOW |
| Use of electrical equipment, shocks and burns | - Keep liquids away from the equipment, check PAT testing, | LOW |
| Harm from Dark box (fingers trapped) | - operate the Dark box carefully | LOW |

## 5. Theory:

### 5.1 The nature of speckle patterns:

A speckle on the resultant interference pattern (a characteristic small dot in the light pattern) has a radius $r_s$ which is dependant on the wavlength, $\lambda$, of the incoming light, the focal length, $f$, and aperture, $d$, of the lenses:

$$r_s = \frac{1.22\lambda f}{d} \qquad (1)$$

This is identical to the 'Airy equation'.

The distribution, $P$, of the intensity, $I$, of the speckle pattern is given by an exponential function and also depends on the average intensity, $\langle I \rangle$:

$$P(I)dI = \frac{1}{\langle I \rangle} e^{\frac{1}{\langle I \rangle}} dI \qquad (2)$$

Note this equation does not work for specular (i.e. follows the law of reflection on a macro scale) surfaces.

## 5.2 The TV Holography Technique:

The direction of deformation to which the interferometer is sensitive is dependent upon the physical layout, and is quantified by the sensitivity vector $k$, which is related to the illumination direction $\hat{k_\mathbf{i}}$ and the observation direction $\hat{k_\mathbf{o}}$ by:

$$k = \frac{2\pi}{\lambda}(\hat{k_\mathbf{i}} - \hat{k_\mathbf{o}}) \qquad (3)$$

An out-of-plane sensitive interferometer can be constructed using a modified Michleson design:

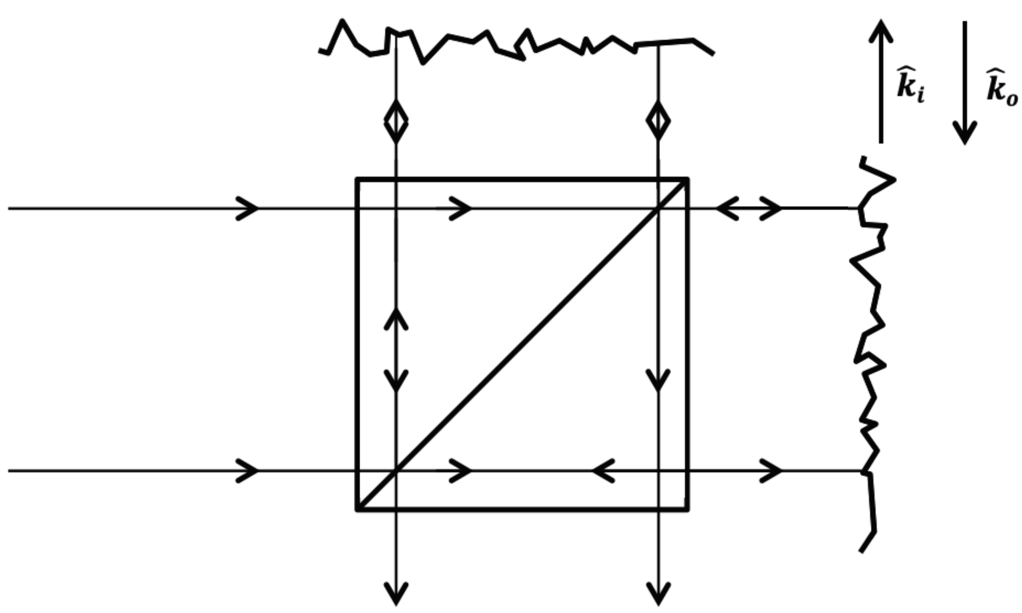



Figure 1: The basic layout for an out-of-plane sensitive interferometer

The phase difference between the object and reference beam is given by:

$$\Delta\phi = \frac{4\pi}{\lambda}(\hat{k_\mathbf{i}} \cdot \mathbf{D}) \qquad (4)$$

where $\mathbf{D}$ is the displacement of a point on the object.

Since the total intensity of the pattern at the observation point is a combination of the two beams ($I_1$, former reflected, $I_2$, latter reflected), we can write it as:

$$I_{\text{before}} = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos\gamma \qquad (5)$$

and after a deformation, given by equation 1, the resultant intensity will be:

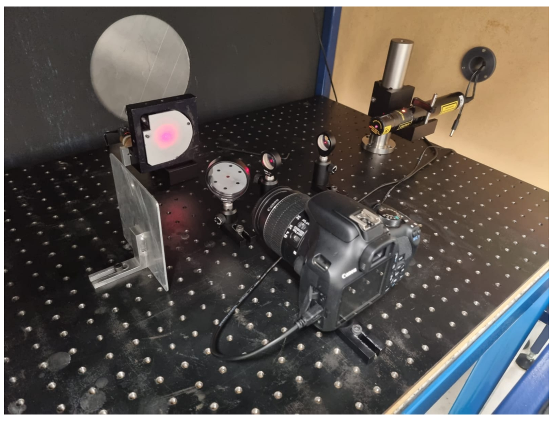
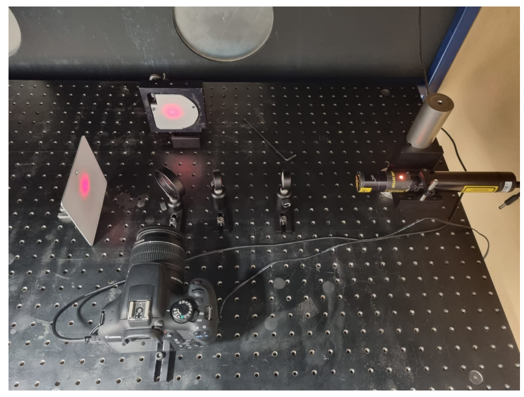
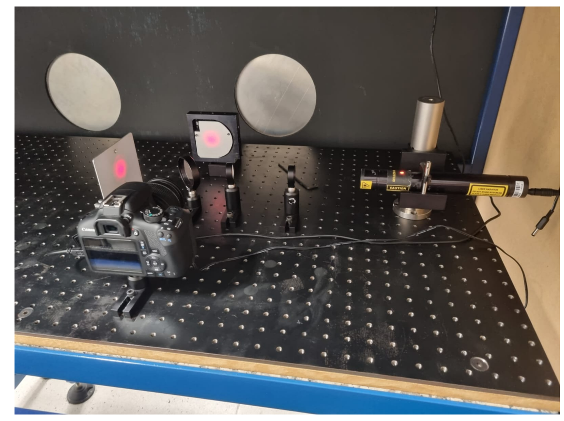
$$I_{\text{after}} = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos(\gamma + \Delta\phi) \qquad (6)$$

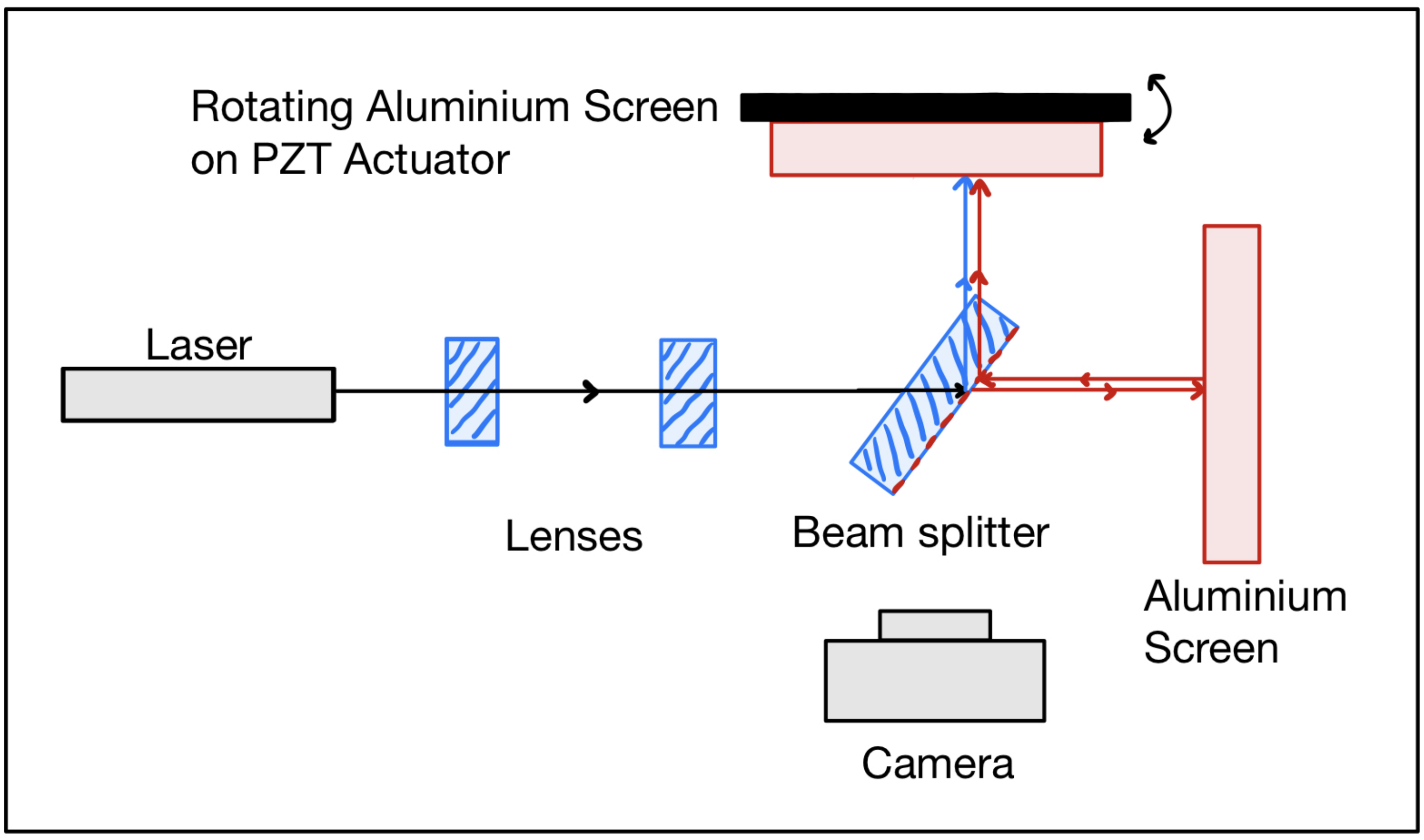
# 6. Experiment:

## 6.1 Tuesday Week 1 27/09/2022:

### 10:52 Initial Setup of Equipment:

We set up the equiupment using reference from the Michleson interferometer design from Wave Optics in Year 2.

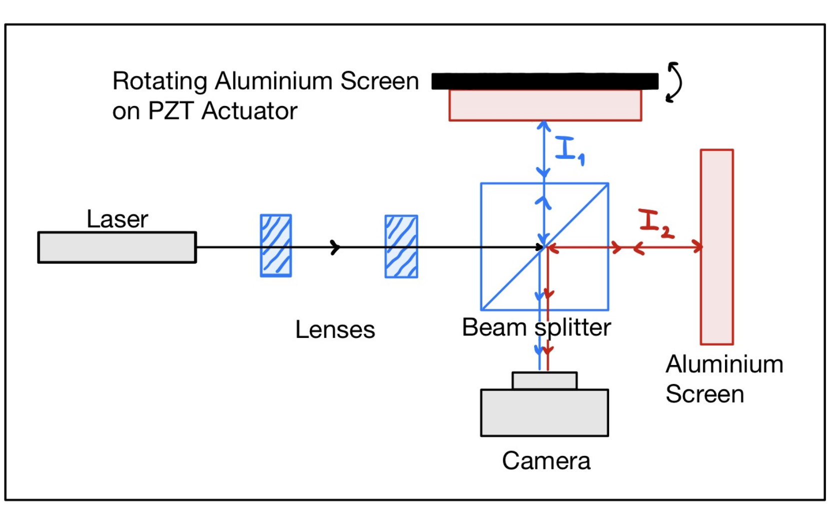







### 15:15 -

On Tuesday, we did not have ImageJ downloaded on the computer connected to the apparatus, not did the Canon camera have the correct driver installed. Thus, we focused more on understanding the equations mentioned in the lab script theory:

- The sensitivity vector $k$ appears to be similar to a wavevector, since it is a unit direction multiplied by $\frac{2\pi}{\lambda}$.
- We think that the illumination direction $k_i$ is the direction of the incoming source onto the aluminium screen (I assume that since the beamsplitter won't necessarily be exactly 45 degrees indident to the laser, that this isnt the same for both screens?), the observation direction $k_o$ is the direction of the resultant light into the camera. This is not necessarily the exact negative of the illumination direction since the screens can be at an angle (not exactly 90 degrees to the illumination direction).




The schematic above is incorrect as the path of the reflected beams are false. The speckle pattern each aluminium screen is there own independant pattern but made with the same coherent source (and so equation 5 has to assume as such).

$I_1$ and $I_2$ have a phase difference of $\gamma$ between them. Once we change the angle of one of the screens using the PZT actuator, the phase difference will further increase by $\Delta\phi$, thus giving us equation 6.

### 16:12 -

Ryand was not able to connect the Canon camera directly into his computer, nor was by any configuration we tried was it able to take a picture. We think that either the wire is faulty, or the camera itself is slightly broken (it still turns on, however it does have a “missing card” error?). Other group’s camera worked immediately when connected to his computer and also took pictures.

### 16:20 -

The Camera was set on auto-focus, thus was changing focus while taking pictures. With it in manual mode, we can manually adjust the focus to the observation plate, and now the taken pictures are saved on Ryand’s computer (c:/Users/yryan/Pictures/”Todays date”).

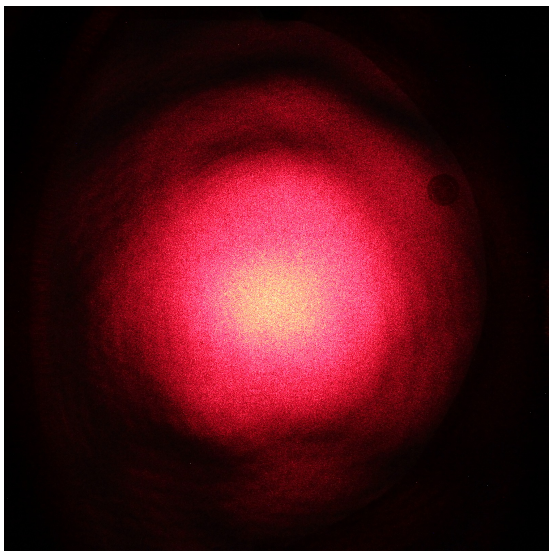
The camera settings are:

- 1”6 Exposure Time
- 1600 ISO
- f16 Aperture
- Auto-wide balance
- Manual focus

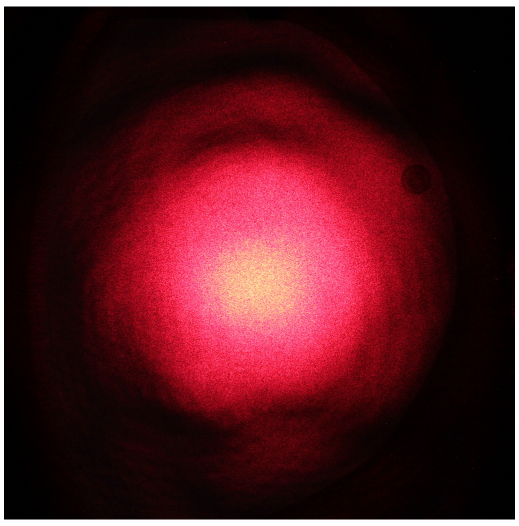
### 16:23 -

Took 3 pictures, one with 0V on the PZT actuator, one with 50.4V, and one with max volts of 99.7V.

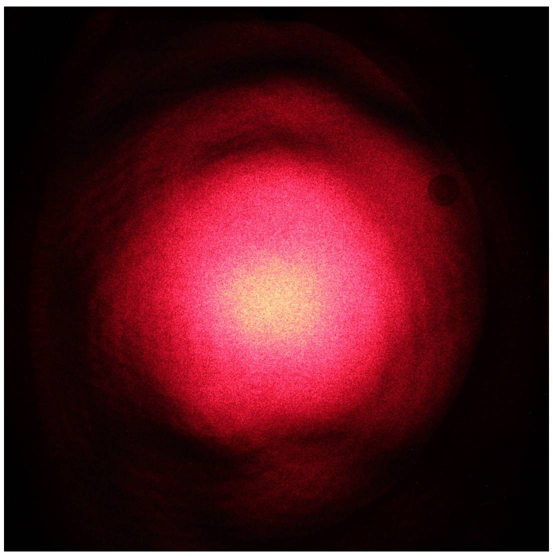
1. Opened 3 images on ImageJ
2. Clicked on process tab
3. Click Image calculator
4. On pop-up, selected 0V image and 50.4V image
5. Selected the “Subtract” option
6. Saved the resultant image as “0000V-0504V.jpeg” (with the process being multuplying the voltage by 1000 to rid decimal point) NOTE: saving as.png does not work, it has to be saved as a .jpeg



THIS IS 0V

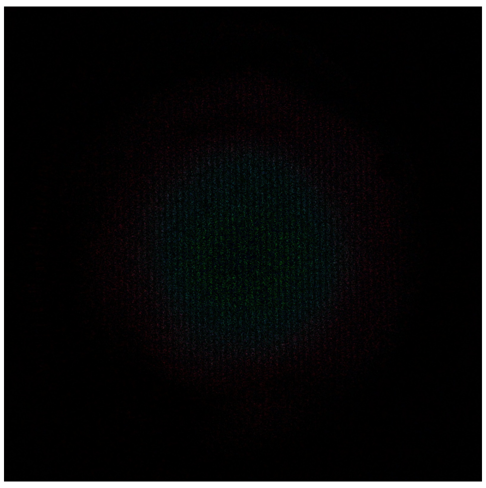


THIS IS 50.4V

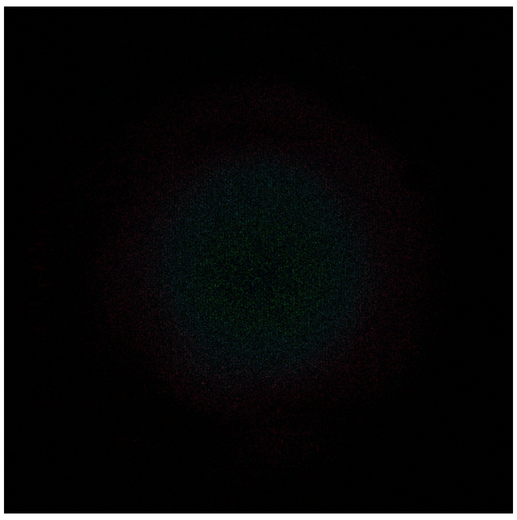


THIS IS 99.7V

The resultant image when subtracting the 0V and 50.4V intensities is:



The resultant image when subtracting the 0V and 99.7 intensities is:



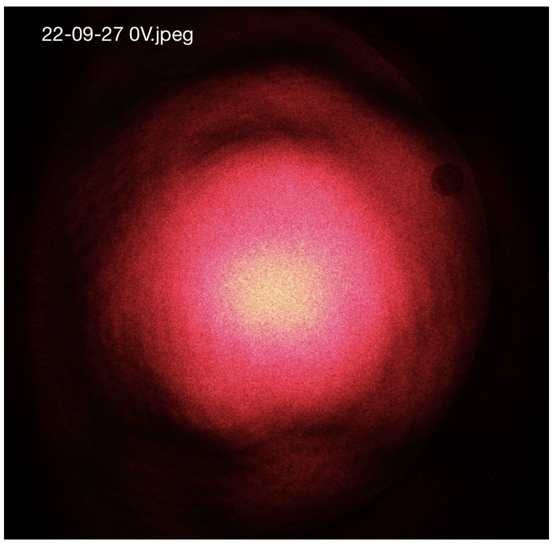
The first image has noticeable vertical lines (fringes), whereas the second appears to contain much more noise.

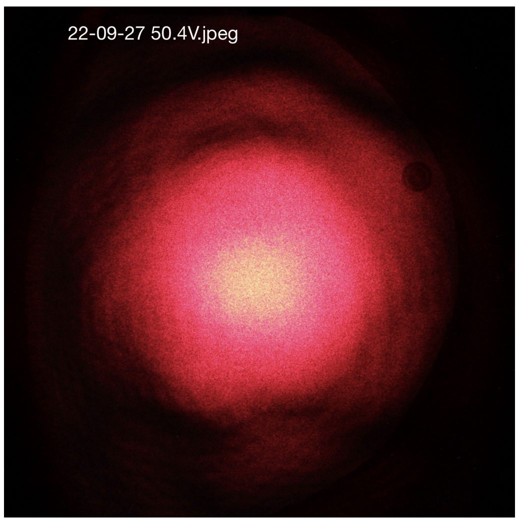
## 6.2 Thursday Week 1 29/09/2022:

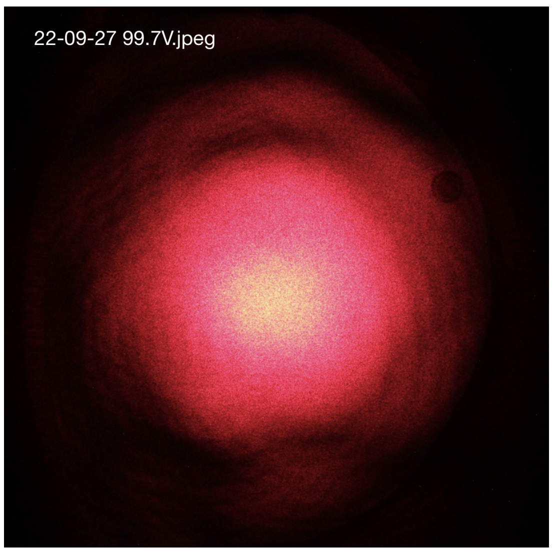
We decided to change the naming scheme for the taken images, it will now be →

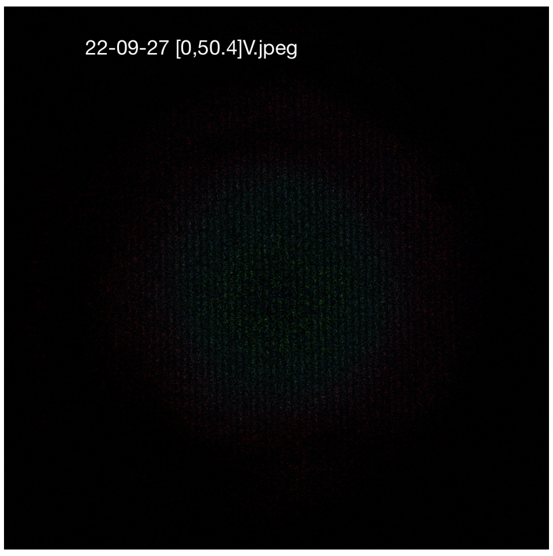
- YY-MM-DD["Voltage before, Voltage after"]V (for the resultant images processed through ImageJ)
- YY-MM-DD "Voltage"V (for the images taken with the Canon camera)

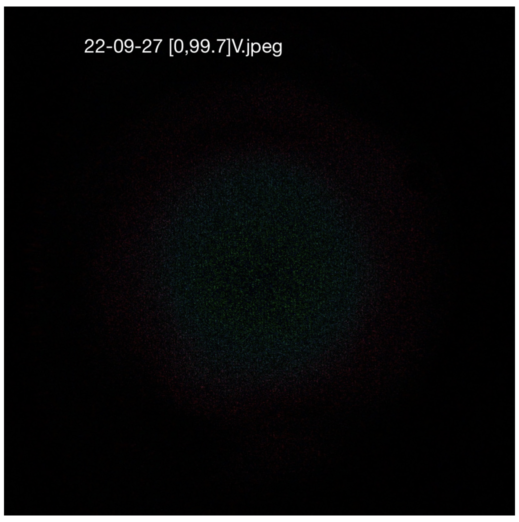
Thus, the images processed before are now:

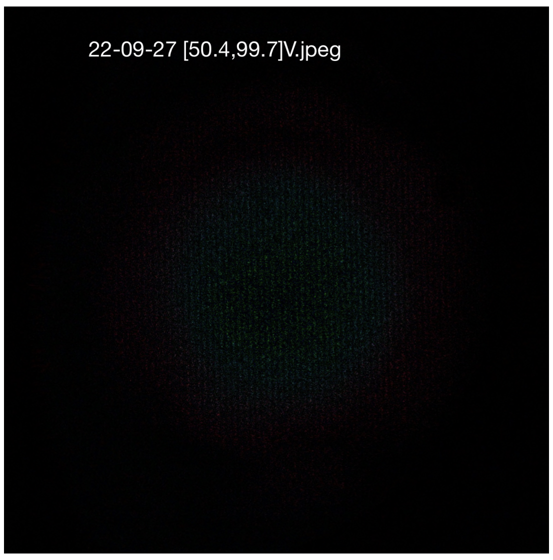












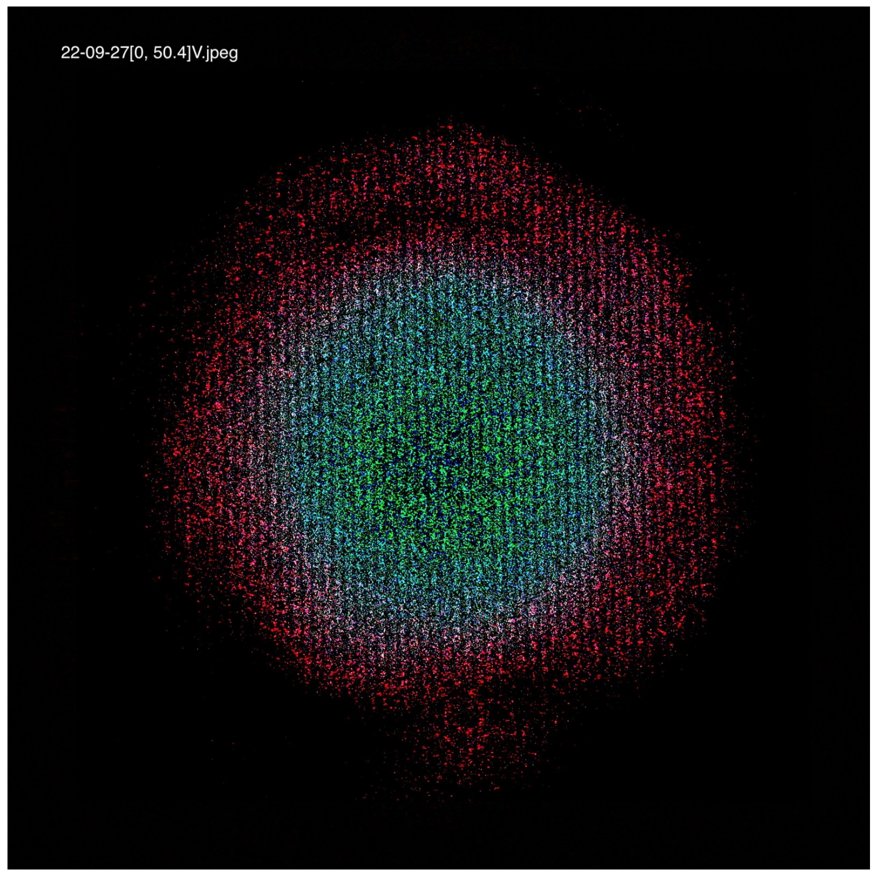

### 9:40-

We did an extra image for the difference between 50.4V image and 99.7V image. It looks very similar to the [0, 50.4]V processed image, seeming to indicate that the characteristics of the fringes (i.e. seperation between fringes) is determined by the difference in the voltage applied to the PZT actuator, and in return a result of the difference in angle of the movable aluminium screen.

In order to make the processed images brighter to more clearly see the fringes, Ryand opened the image on ImageJ and:

- select "Image" tab
- go to "Adjust"→"Brightness/Contrast" which brings up a window
- Press "Auto", and the software automically brightens the image

Thus, for example, the new image for [0, 50.4]V is:




### 9:51-

The goals for today are:

- Collect more images in order to create a rough plot for fringe spacing against voltage in order to make a crude analysis
- Use equation 5 and 6 to derive a relationshp between the angle of the screen to the fringe spacing
- Thus determine how the voltage to the PZT actuator changes the angle of the screen

NOTE: We moved the position of the beamsplitter such that the speckle pattern on the screen on the actuator is not on a nail (the dark spot on the top right of the canon images).

### 9:57-

Retaking canon images with new labelling system, changed position of beamsplitter. We will take a new 0V image as our new reference image and when processing images with ImageJ, all subsequent images will be processed against the 0V image.

Since the wire is not working properly, we need to take a picture with the camera first while plugged into a computer for the computer to register the camera as being connected.

Changed save file path on the Canon application in order to save and rename the file using the naming convention into the correct folder.

### 10:08-

- Taking pictures with all bay doors closed to reduce influence of other light sources
- Moving up by 10V for each image, the voltage appears to remain stable so we are not bothering about how long it takes between capturing images

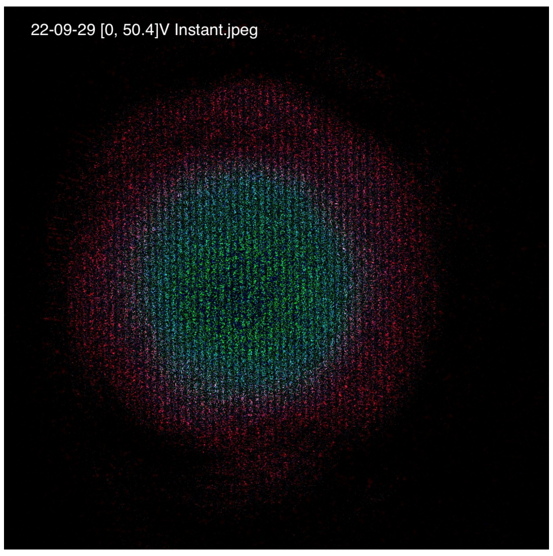
### 10:15-

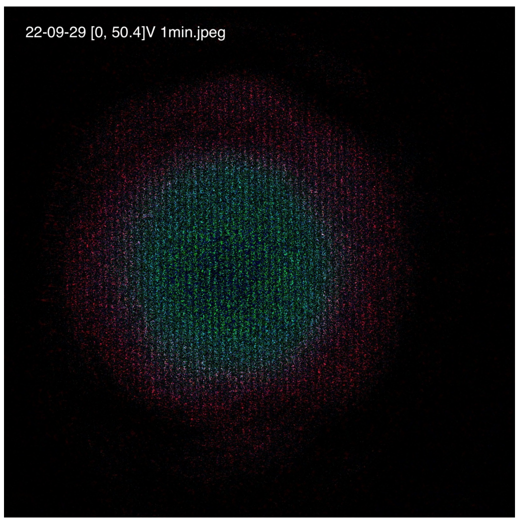
- Once we reached a maximum of 99.8V, we then took images going back to 0V, thus we will save the images as {Before/After YY-MM-DD ”Voltage”V.jpeg} where “Before” means is before reaching the maximum voltage and “After” means after reaching the maximum voltage.

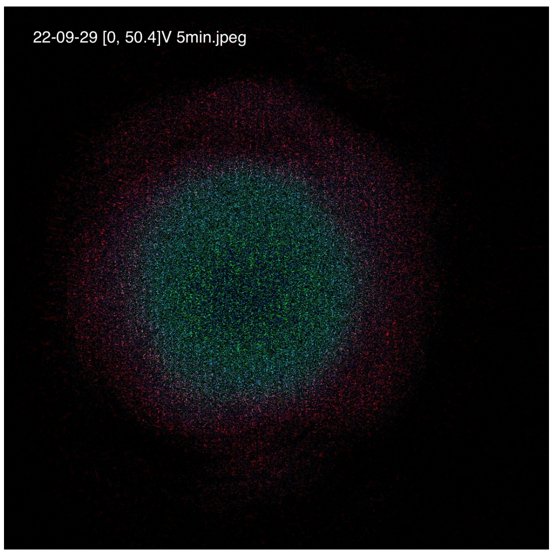
### 11:04-

- All processed images does not appear to have fringes, even the [0, 50]V images, which clearly did have fringes on Tuesday 27/09/22. We retook images for 0V and 50V (was actually 50.6V) and when processed, clear fringes appeared again. Thus we expect that the time between taking images, since before we had a gradual increase to 50V whereas second time round we did it immediately, has an influence on the output.
    - THUS: we will redo 0V and 50V, take one image immediately after changing voltage, one image 1 minute later, and one 5 minutes later to see if tehre is any difference. Making sure not to change anything else about the setup.

- ALSO: just noticed that when taking images in the morning, the lights in the lab were on (no fringes), however now all lights are off (fringes appear),




It is evident that the picture taken instantly after the change in voltage has much clearer fringes than that taken 5 minutes after. Thus, we have decided to:

1. Take smaller voltage intervals to allow the actuator to move a much smaller distance
2. Have longer time periods between changing the voltage and taking the image to allow the actuator to stabilise.

### 11:57 -

We want to check how accurate the camera itself is, and whether we need to change its parameters e.g.:

- ISO (smaller number means less “grainy (noise)” however it becomes less sensitive to light (i.e. worse in dark))
- Aperture size (larger aperture allows more light to enter the camera lense (smaller number means bigger aperture))
- Exposure time (longer exposure time allows more light to enter the camera)

We guess, in order to make the image less grainy, we should decrease the ISO, however increase aperture size and exposure to compensate for worse performance in the dark.

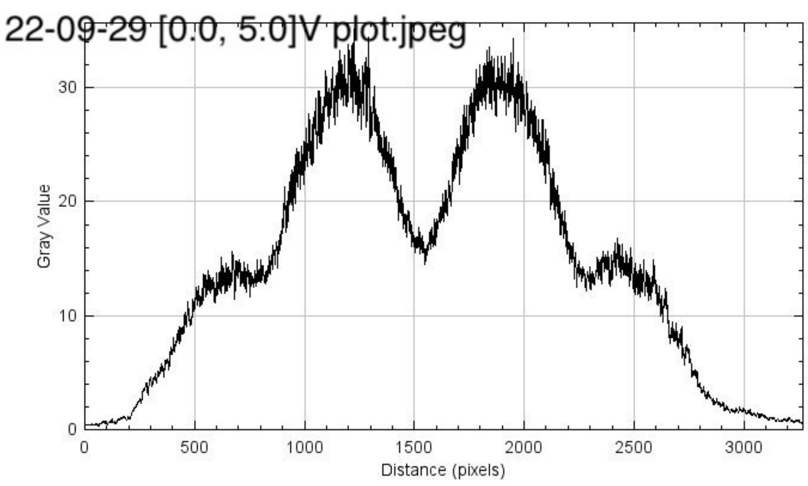
We took two images, both at 0V, but for:

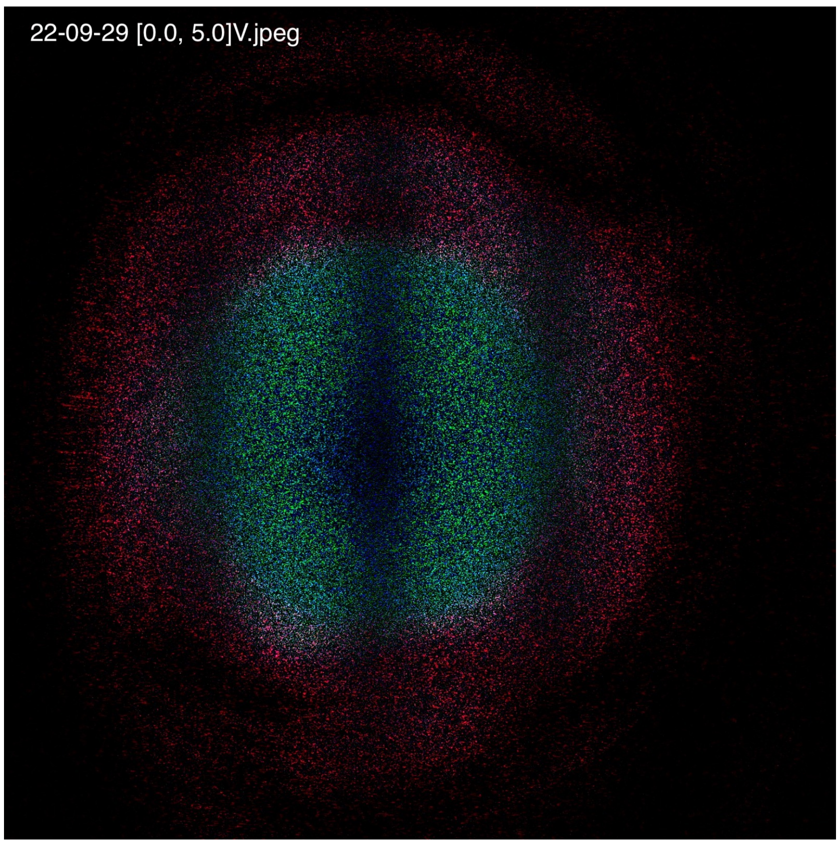
- ISO: 800
- Aperture: F11
- Exposure Time: 3”2

We then took the difference of the two images on ImageJ, and as expected, the result was practically completely black (there was still a little bit of noise).

### 12:10 -

- Redo the data collecting with new parameters. As well as this:
  - Only going up to a max voltage of 50V
  - Interval steps of 5V
  - 1 minute time interval between taking images






The fringes for [0.0, 5.0]V are much larger and are clearer, aligning with our assumptions from earlier.

The plot above was made by drawing a box around the speckle pattern on the left, and then pressing “Cmd+K” on the keyboard to plot the grey value, i.e. “intensity”, against pixel number (from left to right).

NOTE: 50V was still to high a voltage and resulted in an image which had no discernable pattern (no fringes were visible). Thus only go up to a max of 15V from now on.

### 14:01 -

From a brief anaylsis of the plots made from the processed images, one fact that is clear is that as voltage increases, the more peaks are present. For [0, 5.0]V, there are 2 obvious peaks, and for [0, 10]V there are 4 obvious peaks as an example. Other observations we made were:

- The amplitude of the peaks decreased as voltage increased (the curve seemed to flatten)
- It appears as if a gaussian function (the intensity pattern of just the laser) is being convolved with a sinusoidal function.

Our assumption is that as the voltage increases, the sinusoidal function which is being convolved increases in frequency.

### 14:45 -

Ryand found a page on google which encourages us to use an ISO of 200 in the dark, as that will help sharpen the images. We will therefore quickly change the camera settings, redo 0V, 5V and 10V images and compare with the images taken earlier in the afternoon.

### 15:32 -

EVERYTHING GOES WRONG. We tried changing ISO, however initially it appeared as if ISO 200 did not work at all, the output graph for [0, 5]V and [0, 10]V had no peaks. However, when changing back to ISO 800, despite seeming to have everything exactly as it was before, no peaks appeared!

Taking multiple images over and over again, sometimes fringes would appear, but would be off centre and slanted (which has not happened before) and sometimes none would appear at all. What we have decided to do is to redo everything, change lense position to get a solid speckle pattern in the middle of both aluminium sheets, and retake results for [0, 5]V and [0, 10]V for ISO 800.

### 16:13 -

We could not find a solution to the problem, we tried retaking images after cleaning lenses, changing camera angle and adjusting laser angle. We have changed the position of the lenses (MAKE SURE TO MEASURE DISTANCES NEXT TUESDAY) and are retaking data with ISO set to 800 and all other parameters of the camera are the same.

With a quick run of [0, 5]V and [0, 10]V, the final images did have fringes!!!!!! However, they are still slightly slanted. However, before the end of the day, we will take one more collection of images in order to do a more detailed analysis.

**Why are we getting slanted images?????:**

- Even after changing the position of the lenses, we are still getting slanted images, therefore assume it is not a problem cause by the lenses (i.e. a lense was not knocked out of position)
- The most plausible culprit is the PZT actuator, the slanted imaged might be because the actuator is rotating the screen forward:
    - this introduces another change which we originally dont account for. A vertical change also makes sense for a vertical change in fringe seperation

This time, we are going to:

- Going to 20V
- Voltage intervals of 2V

NOTE: The voltage supply is very unstable under 5V, it is very difficult to place to the voltage at exactly 2V or 4V (much easier for 6V), thus for future do not take data points under 5V

Notes during data collecting:

- 2V image was displayed as 2.2V when image was taken
- 4V image was displayed as 4.4V when image was taken

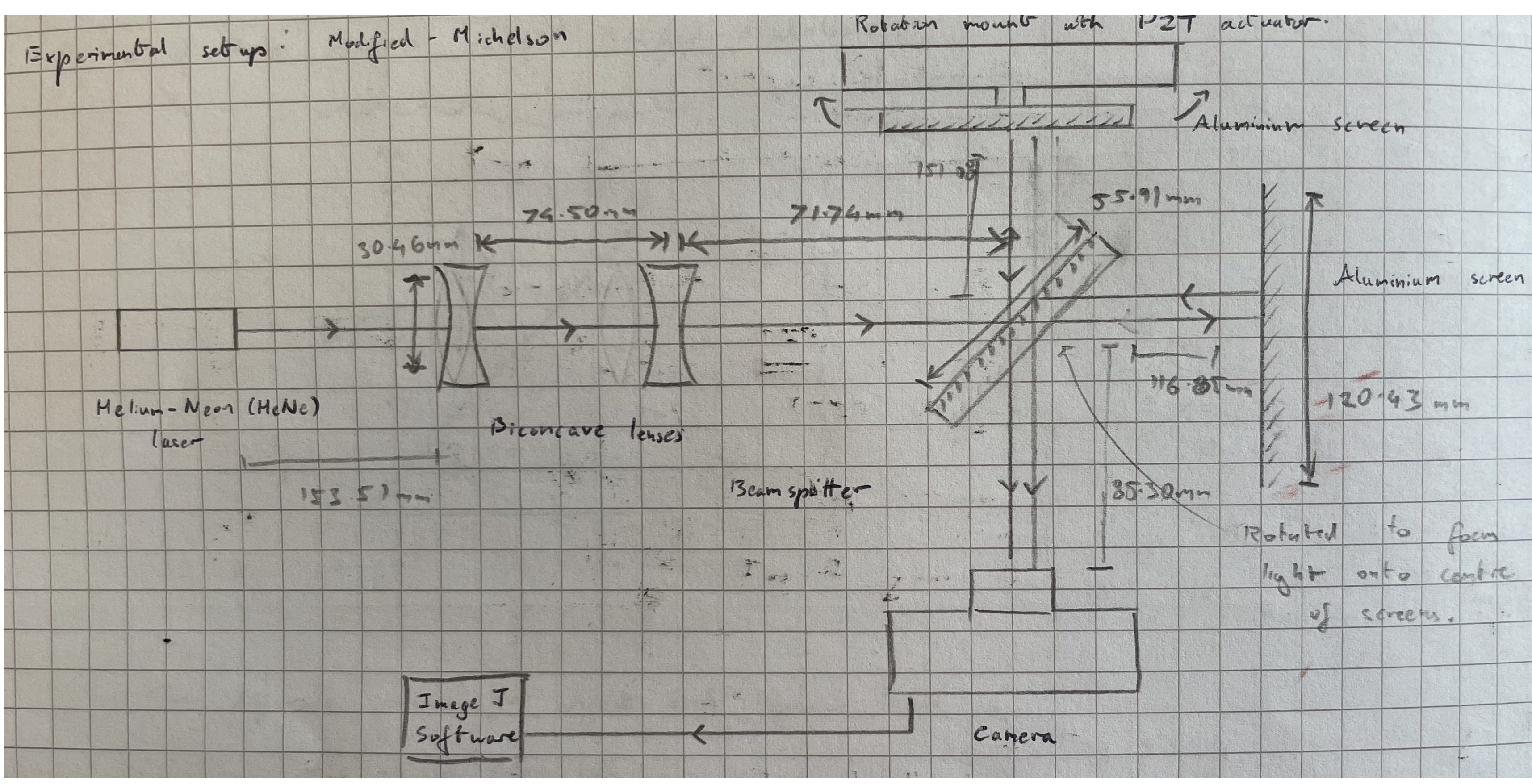
### 16:50 -

AND AGAIN, for seemingly no apparent reason, despite keeping the equipment in exactly the same place after getting fringes, once we start collecting more data, the fringes were not present.

### Goals for next week:

- Make a checklist for parts to adjust and test to see what is the problem
- Understand what is happening with the PZT, see if we can fix it
- If not, ask whether PZT can be replaced
- Before Tuesday, derive an equation which relates phase difference to the two intensity patterns.

## 6.3 Tuesday Week 2 04/10/2022




### 9:03-

**Checklist:**

1. Take 1 set of measurements with equipment in same positions as last week (0V, 10V) and compare with images taken from last week
2. If no result, change camera settings to:
    a. Exposure time - 0.6s
    b. Aperture - F14
    c. ISO: 100 and ISO:800
3. If no desired result, swap PZT actuator with other group and retake measurements
4. If no desired result, adjust lense positioning and orientations to “focus the beam”

### 9:11-

Our assumption is that due to the high voltages we put the PZT actuator under last week and for long periods of time, the actuator got hot (or relatively hot to its operating temperature (if it has one)) and started behaving erratically (i.e. we had slanted fringes, suggesting the screen was rotated in an unexpected direction, or no fringes at all). This would also explain why this problem did not occur on Week 1 Tuesday, since we only had the PZT actuator on for a very short period of time.

Camera settings:

- Exposure time: 3”2
- Aperture: F11
- ISO: 800

### 9:26-

Checklist 1: No fringes present.

Changing camera settings to:

- Exposuer time - 0.6s
- Aperture - F14
- ISO - 100

### 9:31 -

Checklist 2: Fringes are present!

From now on:

- Stick with these new camera settings
- Reset the voltage to 0V after each set of data collection to avoid temperature changes
- In order to make sure that the PZT actuator properly resets itself back to its original position when at 0V, we need to wait much longer (as switching multiple times between 0V and 10V influences the final result).

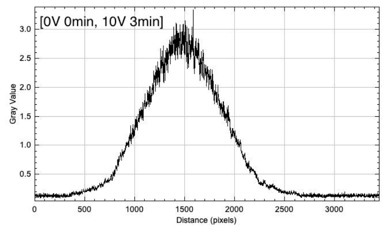
### 9:48 -

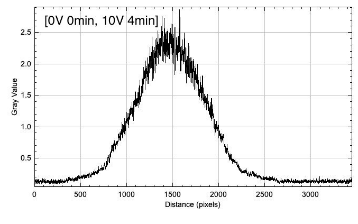
In order to determine the best amount of time to wait between changing the voltage and taking the image, we took images after changing from 0V to 10V after 1 min, 2 mins and 5 mins. However, the intensity of the interference appeared to change over time. Thus, try and figure out what is going wrong we are going to:

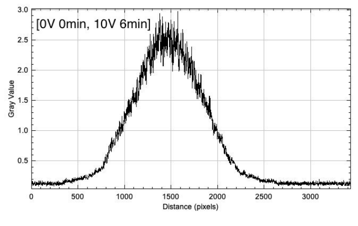
1. Take an image at 0V (the voltage supply was set to 0V for more than 10 mins)
2. Take another image at 0V after 2 mins
3. Immediately change the voltage to 10V and wait 1 minute
4. Take another image after an additional 1 minute
5. Take another image after an additional 3 minutes

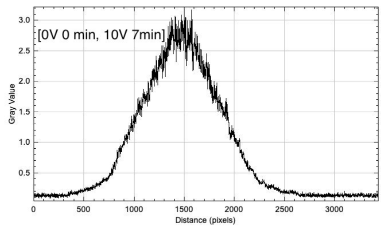
So we would have:

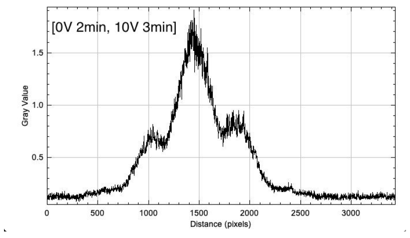
- 2 0V images: 0min and 2min
- 3 10V images: 3min, 4min, 6min and 7min (NOTE: all lights were turned off for 7min image, however there was only one room light on for all other images)

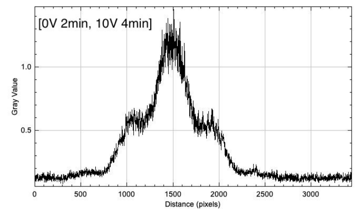


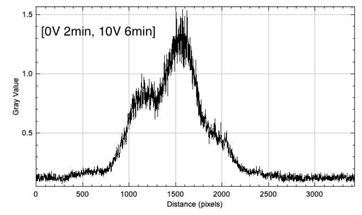



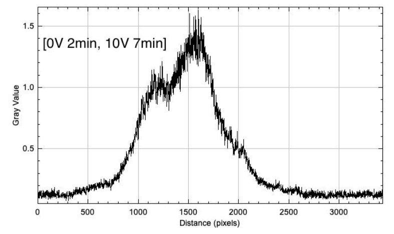









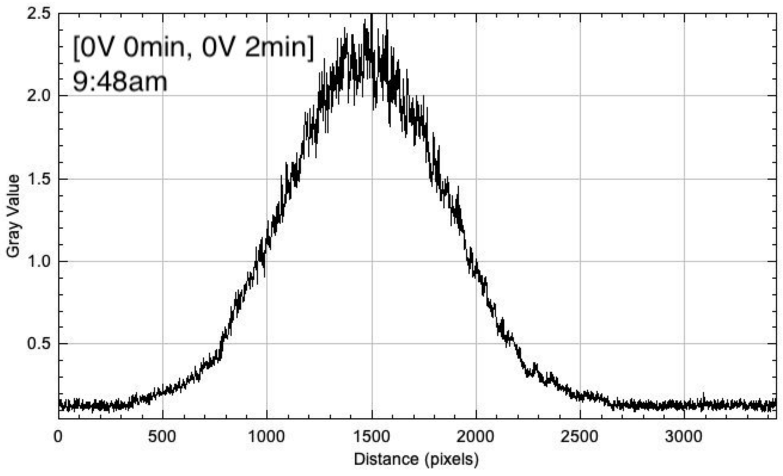


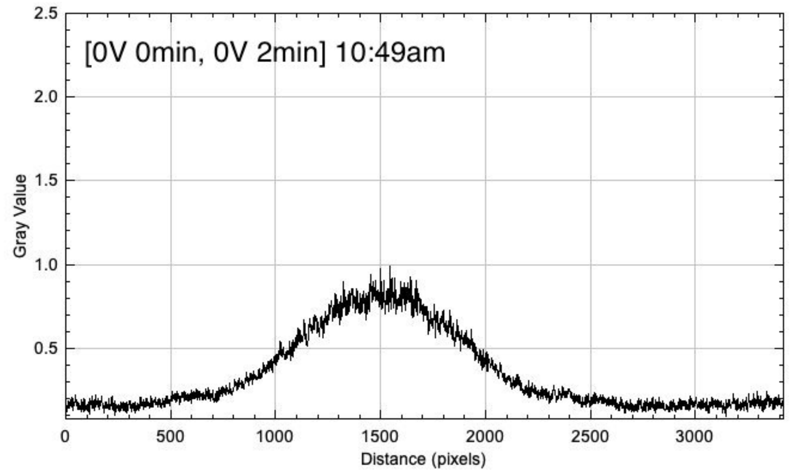

As can be seen by the graphs above, there seems to have been a clear difference between the position of the actuator at 0V 0min and at 0V 2min (and we would expect the [0V 0min, 10V 4min] and [0V 2min, 10V 6min] to be exactly the same).

What we consider is to take 2 images at 0V, 2 minutes apart, and if there is an interference pattern, this suggests that the actuator is still rotatin.

## 10:49-

The voltage supply has been at 00.0V for 45 minutes, we will take two images as suggested above and if there is no interference pattern, we shall change the voltage to 5V and take 4 images: 1min, 2 mins, 3 mins and 4 mins after changing voltage.






in the 10:49 image, the two images are much more comparable, and thus when taking the difference between the two images, the grey value is closer to 0 compared to the 9:48am images. This suggests that the actuator is changing position a lot less once we have waited a substantial amount of time.

The slight hump that is stil present could be from base vibrating during the 2 minutes, the actuator still changing position or a range of other factors, so is very hard to differentiate what causes the most change. We assume that waiting almost an hour between readings should give plenty of time for the equipment to reset, thus I assume it is due to external factors.

## 11:10-

Changing the voltage to 5V and following the procedure mentioned above:

- None of the equipment was changed between the 10:49 data collection and now, so assume that the actutor is in the same position
- The door to the lab is open during this data collection, but was consistently open for all images

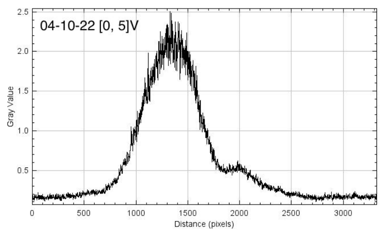
For all of the images taken, the fringes consistently shifts suggesting that the actuatir is still moving after 5 minutes. However, the image taken 1 minute after change in voltage is the clearest. Therefore, we will take images 30 seconds after change in voltage.

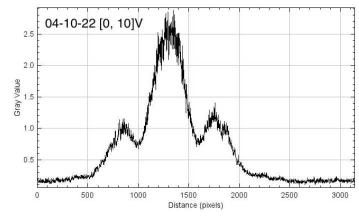
**Assumption as to why this occurs:**

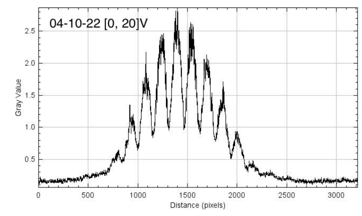
- We assume that the actuator does inded change angle quickly after a change in voltage, however there is a slow drift after the fact.
- Thus having shorter time intervals and being **consistent** will allow us to form a relationship between voltage and fringe seperation, and thus change in angle

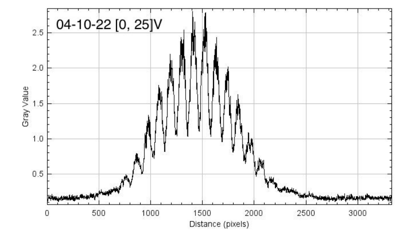
## 11:45-

Starting from 0V, we take images at 5V incremements $\rightarrow$ THIS TIME we will only have 30 seconds between taking each image.










The fringes are much clearer on this round of data collection, the 30 second gap works well. However, with the 5V and 10V graphs, it is slightly difficult to find the exact fringe spacing (especially if we want to do multiple readings), thus we will only start from 15V when taking future data and extroplate down to the lower voltages.

### 13:31 -

**Full experimental procedure:**

- Camera settings:
  - Exposure time: 0"6
  - Aperture: F14
  - ISO: 100

1. Take image of PZT set at 0V
2. Immediately raise voltage by 5V
3. Wait 30 seconds
4. Repeat steps (2.) and (3.) up to 15V
5. Take a picture 30 seconds AFTER reaching 15V
6. Repeat steps (2.) and (3.) up to a voltage of 70V
7. Repeat for decreasing the voltage back to 15V

The additional waits up to 15V is in order to ensure the same behaviour of the actuator for 15V and all other voltages.

### 13:45 -

The graph output of for the images taken when increasing the voltage had very good clear graphs. All of these were used to get a crude approximation of how fringe seperation in pixels is related to the voltage supplied to the actuator.

HOWEVER, when decreasing in voltage, the speckled patterns did not appear to interfere (i.e. no fringes were discernable). What we assumed would happen is the fringe seperation would increase as voltage decreased, however this pattern was not clear on this run of collection.

In order to use the rest of the day effectively, I wrote a barebone python programme to analyse the images for increasing voltage while Ryand collected another set of data using the same experimental procedure described above.

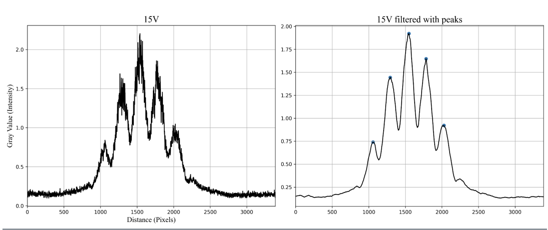
**How does the Python script work:**

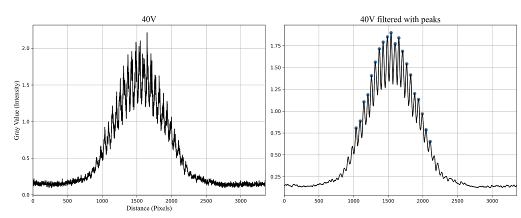
The script is saved on GitHub via https://github.com/stanzi1626/TVHolography. The repository is not public.

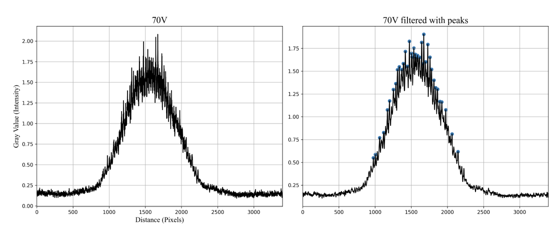
Right now, the script is very barebones and only does the minimum in order to plot a graph for (1 / Fringe Seperation) against voltage. The data points for each graph gathered was saved as a .csv file and saved on '/Users/alexstansfield/Desktop/Lab Images/04-09-22 First run Decreasing/Values/' where they are then loaded in. Here is a quick rundown of how it works:

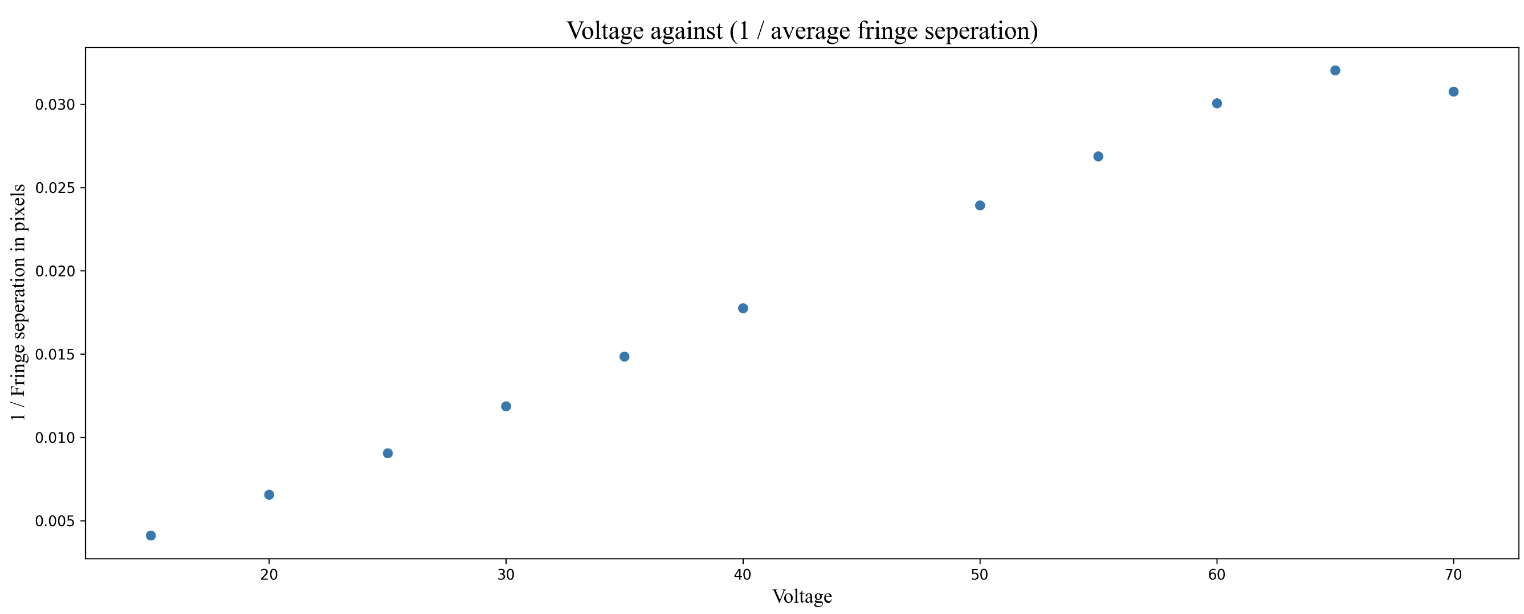
- The `glob.glob` function reads through all files extensions in the folder and `np.genfromtxt` converts the .csv text data and puts into an `numpy.array([])`. All data is saved seperately in a list with the associated voltage (which is used as an identifierm as well).
- Each set of data is passed through a function called a "Savgol Filter" (how the filter works will be mentioned later). This effectively smooths the signal, getting rid of high-frequency noise, and allowing for the `scipy.signal.find_peaks` function to effectively find the peaks of the signal (the function is imported as fp()). The `prominence = 0.1` paramater passed through `fp()` determines how sensitive the function is to peaks, it is *"the minimum height necessary to descend to get from the summit to any higher terrain"* [https://en.wikipedia.org/wiki/Topographic_prominence].
- For each image, there is an associated `savgol_parameter` (which is saved in the dict `SAVGOL_FILTER_PARAMETERS`). This is because the parameter passed through the filter function determines how much the signal is "smoothed". However, for the high voltage images, the signal has a higher frequency, thus the parameter needs to be tweaked such that the behaviour of the signal is not lost.

- NOTE: The savgol parameter needs to be an odd number, the higher the number the “smoother” the signal - thus a smaller number is used for higher voltages
- A plot of both the unfiltered and filtered data is plotted in order to tweak the savgol parameter for each image and the peaks are found for grey values above 0.5 (this is in order not to include the very noise tails on the signal)
- `np.diff()` finds the speration between the peaks at each voltage and an average is calculated using `np.average()`. The inverse of this is then plotted against the voltage to produce the final graph.




NOTE: The blue dots on the plot on the right is the peaks used to calculate average fringe speration.

The final plot suggests that there is a linear relationship between voltage and 1 / fringe seperation. There is a missing point at 45V because the camera did not properly take a picture at that voltage, and when we had realised we had already changed the voltage to 50V. From the 15V and 40V graphs one can see that the noise filter greatly reduces the noise on the signal without losing the overall shape. However, for 70V, since it is a very high frequency signal, it is difficult to discern what is noise and what is from the speckled interference, thus the find_peaks algorithm does not return peaks that it possibly should. Therefore, from now on we shall only go to 60V, as it is the highest voltage which provides an image which clearly still follows the linear relationship.

**How does the `savgol_filter` work?**

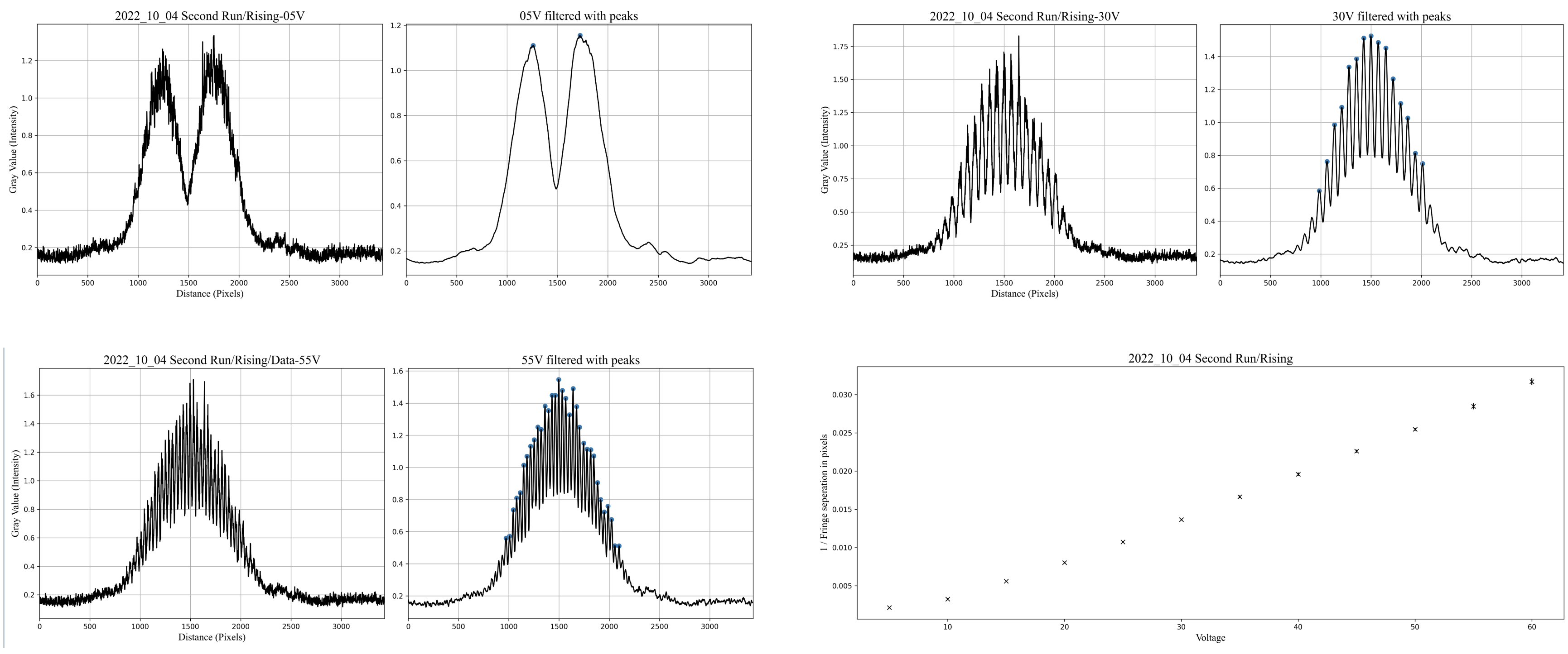
- The full python function is: `scipy.signal.savgol_filter( x , window_length , polyorder)`
  - `(x)` are the y-coordinates of the signal
  - `(window_length)` (which we call the “savgol parameter”) is the length of the filter window. The number of data points that a fit is made to
  - `(polyorder)` the order of the polynomial function which is fit to that data within the window length
- The savgol filter fits a polynomial of order “polyorder” to the data within the “window length”. The window length is then moved across the signal, and for each step a polynomial is fitted until the whole dataset has been filtered.
  - NOTE: We always keep `polyorder=2`
  - When window_length is small to the total number of datapoints, less “filtering” occurs since less of an “average” is being made when there are fewer points. This increases the influence of noise but maintains all of the information of the signal
  - When window_length is comparable to the total number of datapoints, more “filtering” occurs and as such the signal is smoothed more. This reduces the effect of noise but in turn loses information about the signals form.
  - Thus it requires a balance in order to reduce noise effectively while reducing noise. For high-frequency signals (when voltage is high) a smaller window_length is needed otherwise the “peaks” will be “smoothed-out”, whereas for low-frequency signals (when the voltage is low) a larger window_length can be used to reduce noise

## 6.4 Week 2 Thursday 06/10/22:

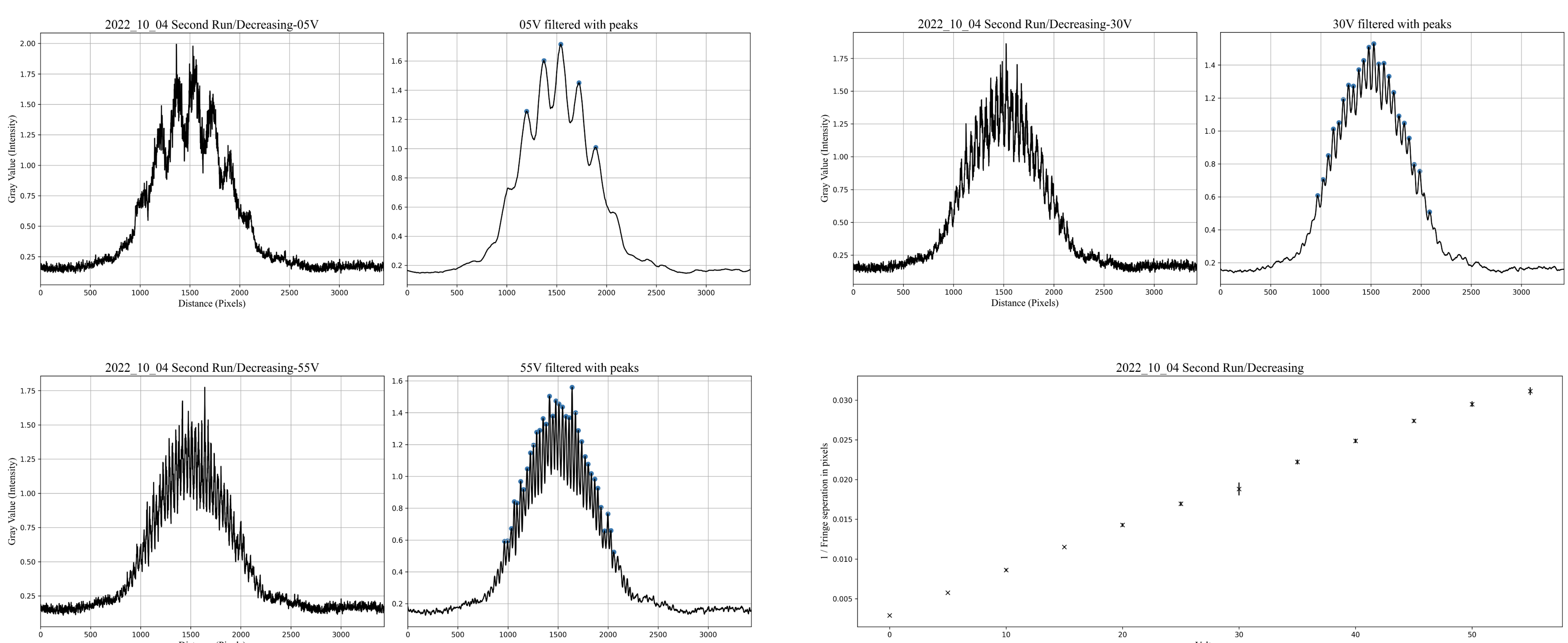
### 10:35-

Used the script made yesterday to analyse the data Ryand gathered on Tuesday Week 2.

Going from 0V to 60V:




Going from 60V to 0V:




I quickly implemented a rough uncertainty calculation for the error on the average seperation. The uncertainties (which appear very small on the final graph) are calculated by:

- using `np.std(peak_diff) / np.sqrt(len(peak_diff))` [where `peak_diff` is the array of the distance between peaks], `np.std()` calculates the standard deviation of the data points, and to find the uncertainty on the average value of the data points is $\sigma_{\bar{x}} = \sigma_x / \sqrt{N}$.
- We think that this understimates the uncertainty since it does not take into account the effect of the smoothing function or the find peak alogorithm to determine the position of the fringes.

There are a few quick notes to make about the differences between the plots of rising voltage and decreasing voltage:

1. It is more obvious with the 5V plot, but it appears as if the decreasing 5V plot looks like the increasing 15V plot and so on. So the gradient of decreasing voltages is different to that against increasing voltages (which i will plot later on).
2. There appears to be less interference when decreasing the voltage, the size of the peaks are smaller than its counterpart when increasing voltage. This is evident in the decreasing 30V plot, which has much smaller peaks than the increasing 30V plot. We assume this is a result of the actuator heating up over the course of the experiment, causing it to vibrate more and as a result the effect of interference is lessened over the exposure time of the camera. However, the decreasing plot appears to still be very linear, whereas with this assumption the gradient should vary

over the course of the collection, whereas in reality it seems to be a sudden change of behaviour once we started to decrease the voltage.

NOTE: Due to the smaller peaks of the decreasing voltage plots, I had to change the prominence value of the `find_peaks()` function to include the very small peaks. Thus:

- Increasing voltge: `peaks, _ = fp(w, prominence=0.1)`
- Decreasing voltage: `peaks, _ = fp(w, prominence=0.02)`

I also adjusted the folders of the github repository so that all of the data and subsequent result plots were stored locally on the repository rather than on my computer to allow for others to interact with the data.

### 13:58 -

Incorporating both increasing and decreasing plots into the script (rather than running the script for each folder sperately) in order to create a plot comparing the two.


In order to obtain the gradient, y-intercept and their associated uncertainties:

1. Import a library called `scipy.optimize.curve_fit` (imported as `curve_fit`)
2. There are 4 inputs when `curve_fit` is called:
   a. desired fitting function (just the linear function y = mx + c in this case)
   b. x-data, y-data and finally the uncertainties along the y-axis (no implementation of uncertainties along x-axis as of yet)
3. The expected values and uncertainties for m (gradient) and c (y-intercept) are returned

NOTE: We only fitted the lines of best fit emitting the first and last two data points

## 6.5 Week 3 Tuesday 11/10/22:

### 9:21-

Looking at the final plot obtained on Week 2 Thursday, there are some interesting observations that we made:

- As expected, the gradients of the to lines of best fit are very similar (although the uncertainties are small for the reasons mentioned last week). This implies that the relationship between fringe seperation and voltage is independent of whether the voltage is increasing or decreasing
- However, the y-intercepts are the negative of each other. This implies that if there is a mechanism which causes this offset from the origin (i.e. the actuator taking time to rotate) that its impact on the behaviour of the system flips once the direction of voltage chaneg flips. This observation appears to fit this assumption that the actuator does indeed continue to slowly rotate after a change in voltage.
- Finally, there are the signs of a hysteresis. The first two and last two points for both rising and decreasing voltage appear to “flatten out”. This is why we did not fit the line of best fit using those data points

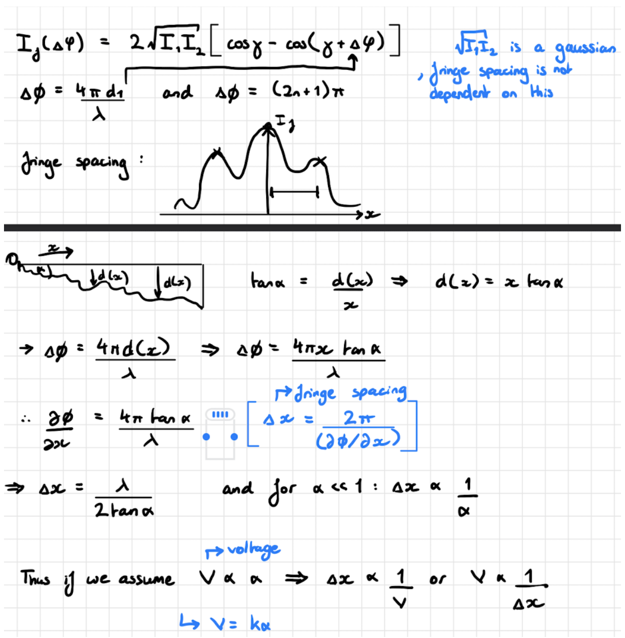
### 10:09-

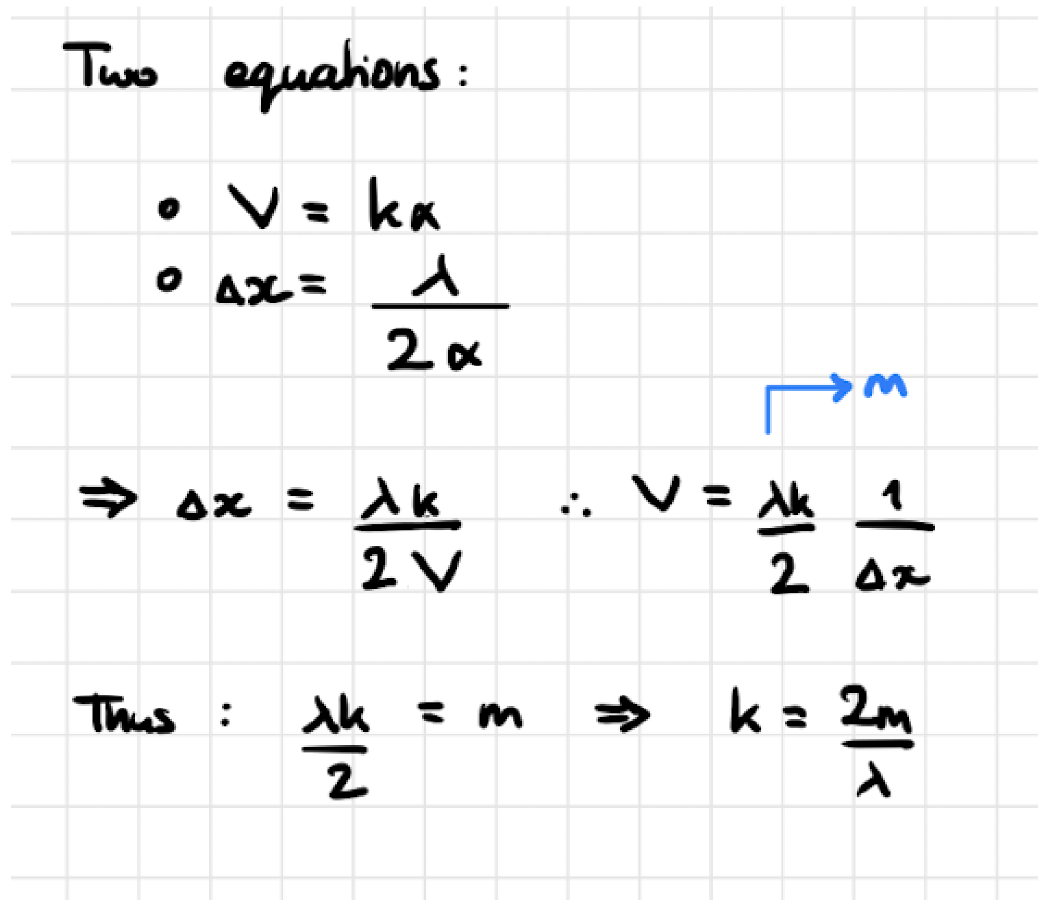
### AIMS:

- Understand and derive equations for the behaviour of the interference patterns
- Better determine the sources of uncertainty in the epxeriment and try to implement them
- Collect more data
- Understand the limitations of the experiment

### 10:30-

This is the derivation I made to relate the fringe spacing,$\Delta x$, to the angle of the actuator, $\alpha$, and thus the voltage to $\alpha$ by the constant of proportianality $k$.




The equation for the phase difference between two coherent waves is:

$$\Delta\phi = \frac{2\pi}{\lambda} \times D, \tag{7}$$

where $D$ is the path difference between the two waves. From the traingle drawn in the pdf above, it is clear to see that $D = 2d(x)$ where $d(x)$ characterises the deformation of the surface. Since for this simple case where a “flat” (it is still non-specular) screen is being rotated by an angle $\alpha$, the relationship between the displacement $d(x)$ and the distance along the screen, $x$, is:

$$d(x) = x \tan\alpha. \tag{8}$$

From (”https://www.cambridge.org/core/services/aop-cambridge-core/content/view/121E485E252754101A090326A71566BD/9780511622465c3_p122-164_CBO.pdf/speckle_pattern_interferometry.pdf”), we find that the fringe spacing is proportional to the inverse of the rate of change of the phase difference across the screen:

$$\Delta x = \frac{2\pi}{(\partial\phi/\partial x)}. \tag{9}$$

Combining equations (7) with (8) and then substituting into equation (9) gives:

$$\Delta x = \frac{\lambda}{2\tan\alpha} \tag{10}$$

and for small $\alpha$, we can approximate $\tan\alpha \approx \alpha$. Thus we get that $\Delta x \propto 1/\alpha$ which is the relationship we got at the end of Week 2. Furthermore, if we assume that the voltage supplied to the actuator, $V$, is linearly related to the angle of the actuator, $\alpha$, i.e. $V = k\alpha$, where $k$ is the constant of proportionality, we can sub this into equation (10) to get:

$$k = \frac{2m}{\lambda} \tag{11}$$

where $m$ is the gradient of the line of best fit of the plot (1/fringe space) against Voltage.

### 13:31-

We spent the rest of the day rejigging the python script in order to quantify the errors in the experiment.

Code crashed, so will attempt to fix over the week and will jot down notes next week Tuesday

## 6.6 Week 3 Thursday 13/10/22

We will have the same aims as we didn on Tuesday.

### 9:50-

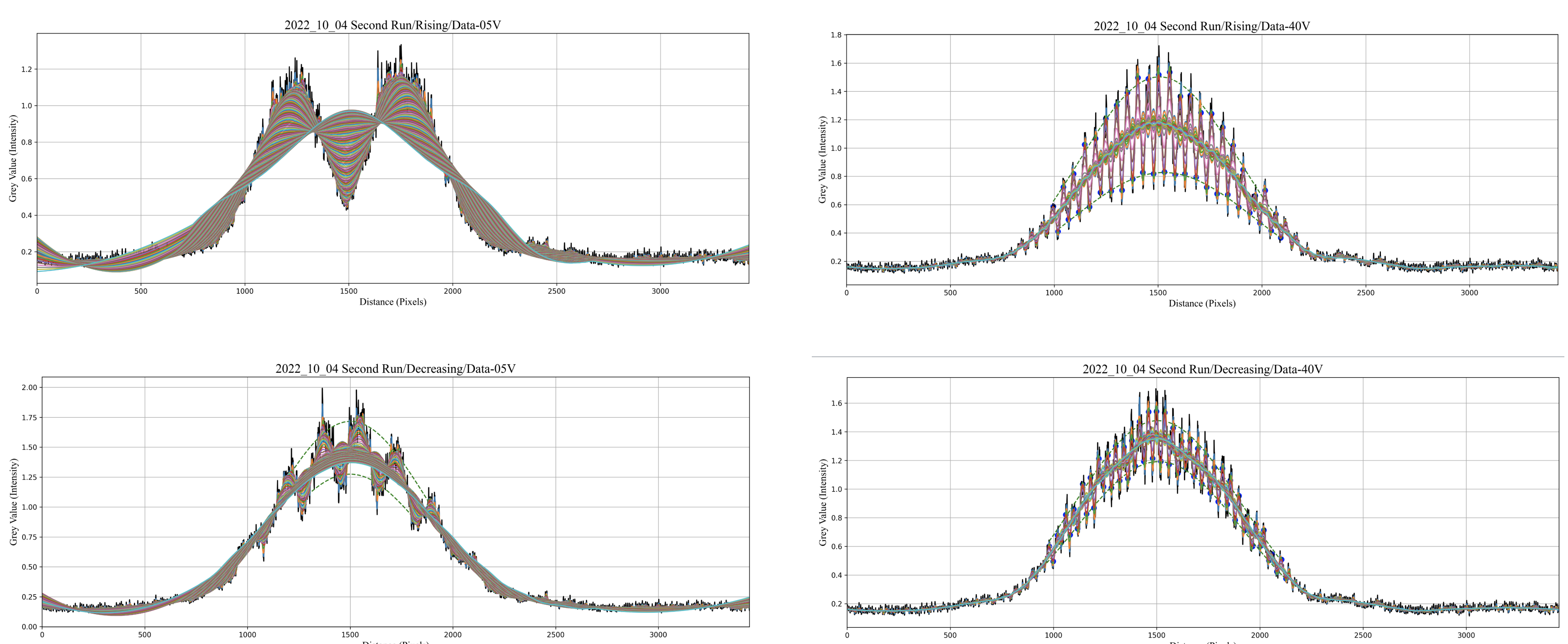
Ryand took more data using the same method, however this time went to 100V to better understand the limitations of the experiment.

### 10:10-

My plan to quanitify the errors on the filters we used to find the peaks (and troughs) of our data points is to use a method to show how well the smoothed data (using either the savgol filter or gaussian convolution) fits with the unfiltered, noise data (for example using the chi-sqaured test).

We can then adjust the parameters used in the smoothing algorithm, such as the `window_length` in the savgol filter, to find a range of parameters which gives a reasonable fit to the unfiltered data. For each paremeter, we use the `find_peaks` function, creating a set of different but (hopefully) close peaks. We can then either take $\mathrm{range}/2$ as the uncertainty or use `np.std()` to find the standard deviation.

I also fitted a gaussian curve to the peak and trough points, this will be in order to quantify the “visibility” of the fringes.




The plots above show what the smoothed signal would look like for `window_length` from `range(11, savgol_0*10, 10)` where savgol_0 is the original savgol parameter we thought looked the best for that set of data. As expected, when the savgol parameter is very large, i.e. the fit is made with lots of data points, the smoothed signal loses almost all information about the behaviour of the original noisy signal. However for very small savgol parameters, the smoothed curve is very influenced by the noise in the signal and thus multiple very small peaks not corresponding to the fringes are found.

The dotted green like is the fitted Gaussian on the peaks and troughs.

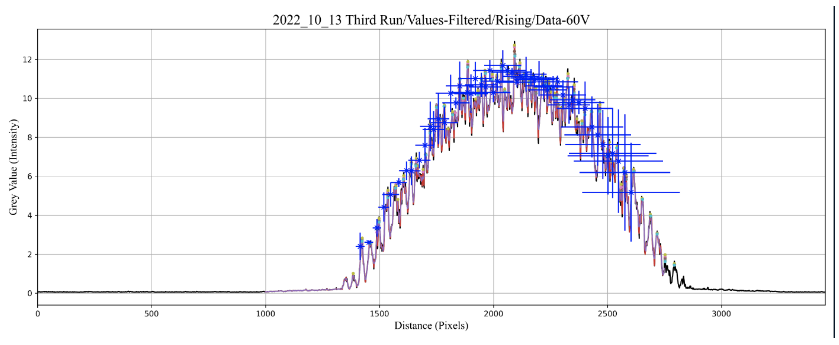
NOTE: that it is only fitted if the number of peaks and troughs is greater than 3 (otherwise the fitting function will not work).

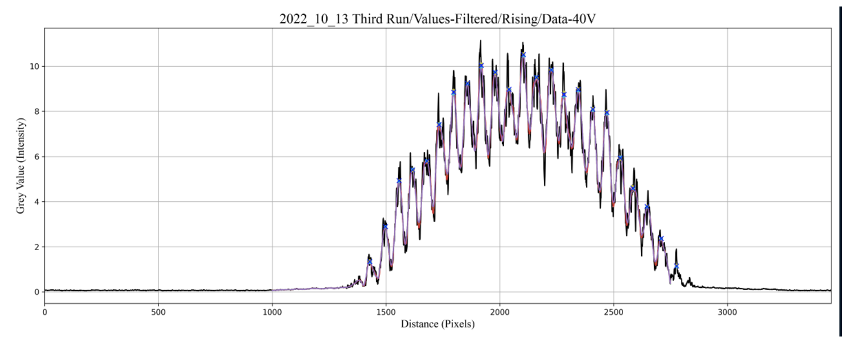
## 6.8 Week 4 Tuesday 18/10/22

### Problems with quanitying errors using different Savgol Parameters:

On GitHub, there are now two “main” branches:

- the “master” branch which includes the code written for the method mentioned on Week 3 Thursday
- a branch called “Mon-17-oct-reset” which builds on the code which was written before Week 3 Thursday




The main problem when finding the peaks for graphs produced using the Savgol filter with different window lengths is that, as seen by the 60V graph produced above, is that the number of peaks found can be different. This then makes it very difficult to compare the x-coordinates of each found peak, since for some of the graphs a peak would not be found at all and would most likely compare for a completely different peak.

This is evident by how the erros become very large for larger distances, the error is including peaks at a completely different distance.

I tried to further filter the peaks by calculating the mean and standard deviation of a set of x-coordinates, and then seeing whether a point was further than 3 standard deviations from the mean, however this does not filter all of the points. Also, at that point it would be more effective to just fit a gaussian to each peak (THIS COULD BE A GOOD IDEA TO FOLLOW)

HOWEVER, this method only appears to break down at around 40V. This is the point where the fringe spacing is too small for the algorithm to be able to differentiate peaks effectively for different parameters. Thus:

This method will only be used up to 40V.

MOREOVER, the additional error found using this is much smaller than the error on the set of fringe spacings themselves.

Thus, we propose two new possible ways of finding the error which will also propogate through more effectively (i.e. create a slightly larger error close to that which we found by just looking at the fringe spacings by eye):

1. Plot the different graphs at different savgol parameters and then check ourselves how much we think the peak deviates
2. or (as mentioned above) fit a Gaussian to each peak and then calculating the $\sigma$ value of that peak, since we expect 67% of the time the true peak to lie within that range, that will be our uncertainty

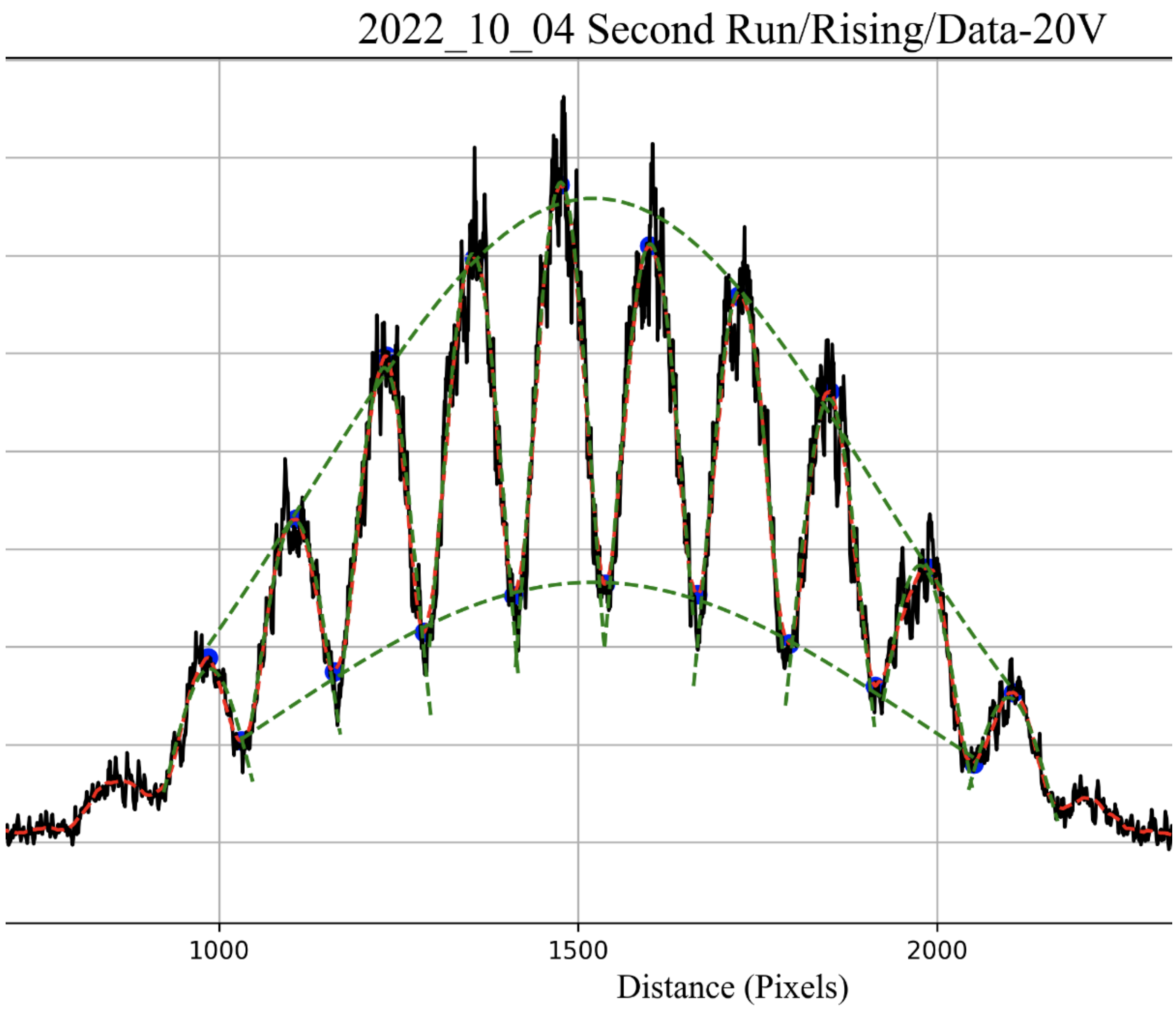
Today, I will follow on with the “Mon-17-oct-reset” branch following method (2).

## 9:55-

Plan of how I am going to implement the gaussian fits to each peak:

1. Find savgol parameters where the output graphs effectively find the peaks I am happy with
2. Calculate the average fringe spacing
3. Use this average fringe spacing as the range around the peaks to fit the gaussian to (in order to not include other peaks)

## 10:43-




Without having to change the steps described, the output (like for the 20V rising plot shown above) appears to have fitted a gaussian to each peak effectively. Since the `scipy.optimize.curve_fit` also produces a covariance matrix, we can find the uncertainty on each parameter as well if we want to.

NOTE: It is much better to use the covariance matrix for the uncertainty of the mean (i.e. the peak) instead of using the sigma value of the fitted gaussian.

### 12:23-

***Final process:***

- Using `scipy.filter.savgol_filter` the data is smoothed with the `window_length` being determined by eye and using `scipy.signal.find_peaks` to find the peaks of the smoothed signal
- Filter the peaks to those only above 0.5 to not include background noise outside of interference pattern
- Find the average difference between peaks (has a variable name of `fringe_spacing_guess`)
- Fit a curve using `scipy.optimize.curve_fit` which is a function of `A * np.exp((x_data - mu) ** 2 / (2 * sigma ** 2))` (i.e. a gaussian), where mu is the mean (i.e. the peak), to each peak. This is done with the raw data for x-ccordinates within the range peak $\pm$ fringe_spacing_guess / 2.
- `scipy.optimize.curve_fit` returns two matrices
    - `param` which is a 1D array with the expected values for: A (multiplicative constant), sigma and mu
    - `cov` which is a 2D covariance matrix with the form:

$$\text{cov} = \begin{pmatrix} \sigma_A^2 & \sigma_A\sigma_\sigma & \sigma_A\sigma_\mu \\ \sigma_\sigma\sigma_A & \sigma_\sigma^2 & \sigma_\sigma\sigma_\mu \\ \sigma_\mu\sigma_A & \sigma_\mu\sigma_\sigma & \sigma_\mu^2 \end{pmatrix}$$

NOTE: subscript $\sigma$ refers to the standard deviation in the Gaussian function

- Thus the new x-coordinate of each peak is: `param[2], np.sqrt(cov[2,2])` = $\mu \pm \sqrt{\sigma_\mu^2}$
- We now have a list of positions for the peaks with their associated uncertainties, stored in `mean_peak` and `sigmas`.
- We get the fringe spacings with `np.diff(mean_peak)` and their uncertanties are added in quadrature, both being saved in `fringe_spacing` and `fringe_spacing_uncertainty`:

$$f = x_2 - x_1$$
$$\sigma_f^2 = \frac{\partial f}{\partial x_1}^2 \sigma_{x_1}^2 + \frac{\partial f}{\partial x_2}^2 \sigma_{x_2}^2$$
$$\sigma_f = \sqrt{\sigma_{x_1}^2 + \sigma_{x_2}^2}$$

- We can then take the weighted mean to find the average fringe spacing:

$$\bar{x} = \frac{\sum_{i=1}^{n} \left(\frac{x_i}{\sigma_i^2}\right)}{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}} = \frac{\sum_{i=1}^{n} (x_i \cdot w_i)}{\sum_{i=1}^{n} w_i}, \qquad w_i = \frac{1}{\sigma_i^2}.$$

$$\sigma_{\bar{x}} = \sqrt{\frac{1}{\sum_{i=1}^{n} \sigma_i^{-2}}} = \sqrt{\frac{1}{\sum_{i=1}^{n} w_i}},$$

These images for the formula of the weighted mean are taken from Wikipedia: https://en.wikipedia.org/wiki/Weighted_arithmetic_mean

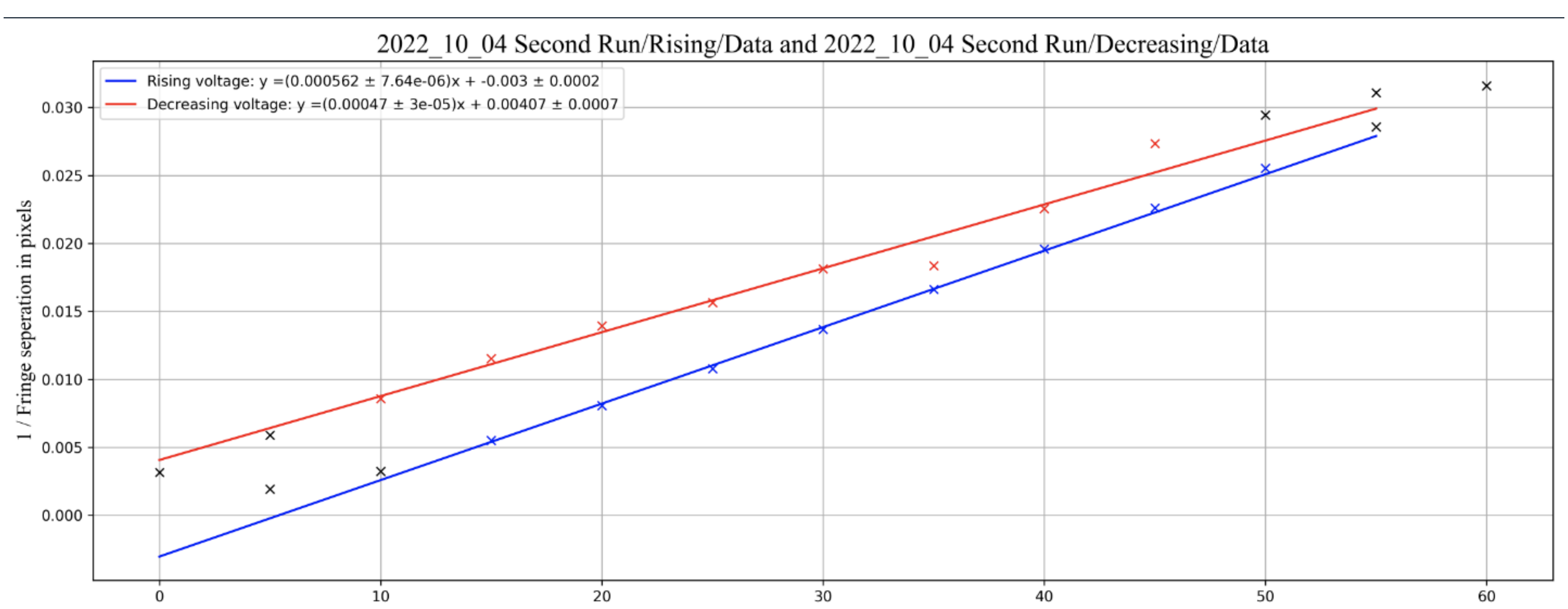
- which in TVHolography.py is written as:

```
def weighted_arithmetic_mean(data, uncertanties):
    weighted_uncertanties = np.power(np.power(uncertanties, 2), -1)
    weighted_mean = np.sum(weighted_uncertanties * data) / np.sum(weighted_uncertanties)
    weighted_mean_standard_error = 1 / np.sqrt(np.sum(weighted_uncertanties))
    return weighted_mean, weighted_mean_standard_error
```

- Since we are plotting 1 / fringe spacing, the uncertainty is then:

$$f = \frac{1}{x}$$
$$\sigma_f^2 = \frac{\partial f}{\partial x}^2 \sigma_x^2$$
$$\sigma_f = \frac{1}{x^2}\sigma_x$$

- We can then plot this as usual against Voltage. For “2022_10_04 Second Run” the resultant plot is:

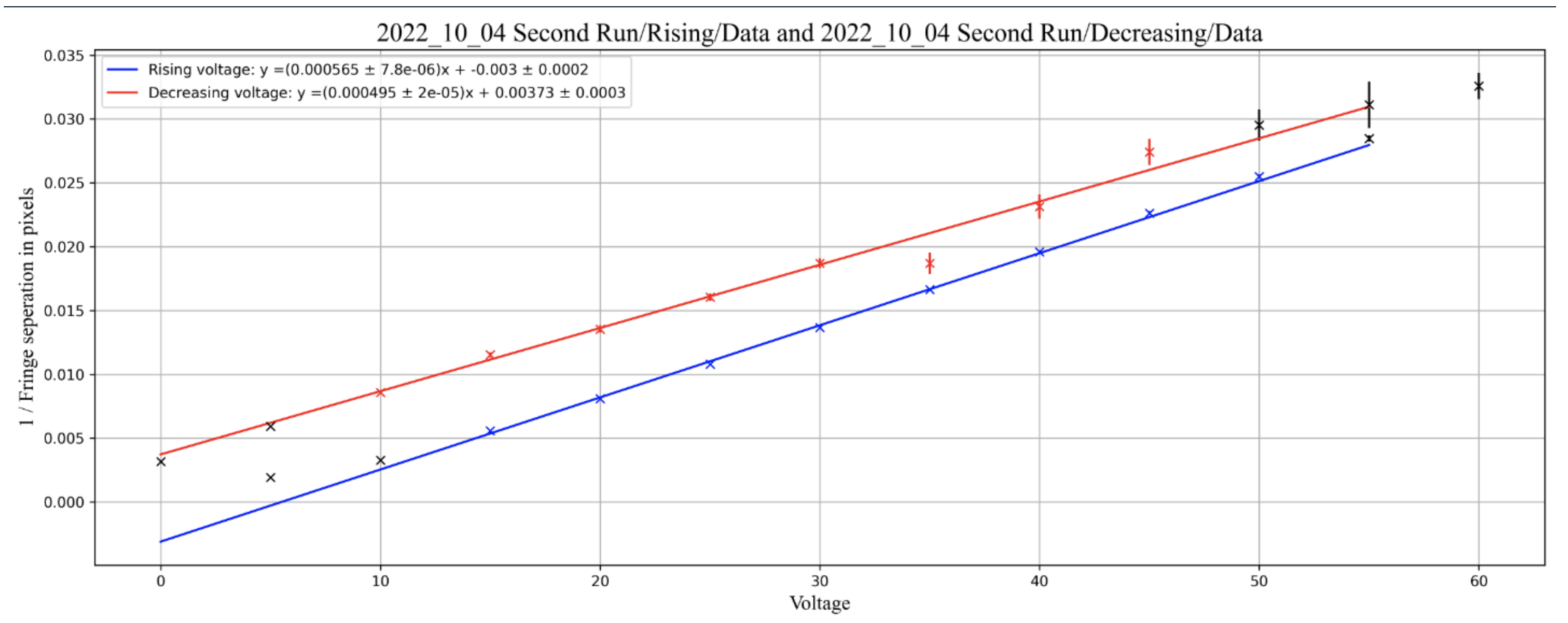



It is clear to see that with this method, by taking the weighted mean, the uncertainties of the data points are much smaller, to the point that they can hardly be seen. This is because the weighted mean of the fringe spacings has a standard error of $\approx 0.2\%$, which is too small. It does not effectively propogate the errors from the difference in fringe spacings because the uncertainty of each peak is so small.

If, however, we just take the uncertainty of the average to be:

$$\bar{\sigma} = \frac{\sigma}{\sqrt{N}}$$

where $\sigma$ is the standard deviation of the fringe spacings and $N$ is the number of fringes - 1, we get:

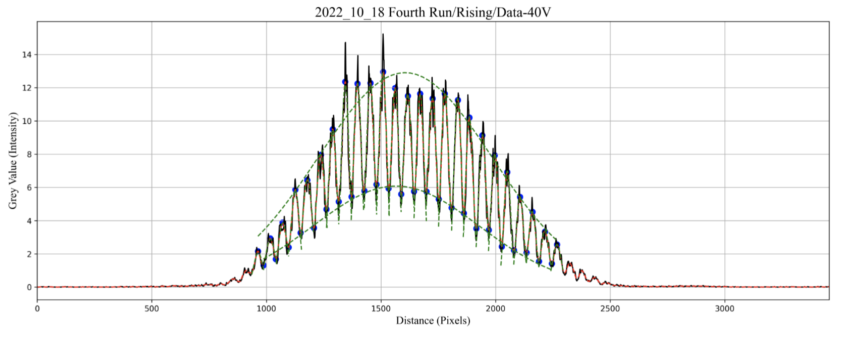



NOTE: This is the "same" graph as Week 2 Thursday. However, they are not exact most likely due to different savgol parameters (in 35V increasing it was clear to see that some peaks were missed) and different find_peak prominence.

The uncertanties for this plot are much bigger, especially the 35V increasing point which is evidently an outlier now has a larger uncertainty. Moreover, at higher voltages, as the algorithm breaks down and starts to miss peaks or add "phantom" peaks, the uncertainty would be expected to be larger which is the case.

### 14:42-

Ryand took a new set of data, now with an aperture of F5.6 (theorising that a larger aperture would increase visibility). This is called "2022_10_18 Fourth Run" in the github respository.





NOTE: In the plot above, we plotted the graph ***after*** having changed the contrast of the image. This effectively multiples the grey values by some amount and allows the script to more effecticely find peaks

In an attempt to increase visibilty, Ryand increased the aperture size of the camera. We think that this allows more of a difference between peaks and troughs.

### 15:16-

As extenstion, we decided to see what the interference pattern looked like when we heat up an aluminium block.

- First we took a reference picture at room temperature
- Then, using a heat gun and an infrared thermometer, we heated the aluminium block to around $80°$C.
    - Since this technique measure the ***deformation*** of the reference material w.r.t. a reference image, we focused the heat gun at the front face of the aluminium block
    - We theorised that this would cause the surface being heated to buldge outwards (since metals expand when heated)

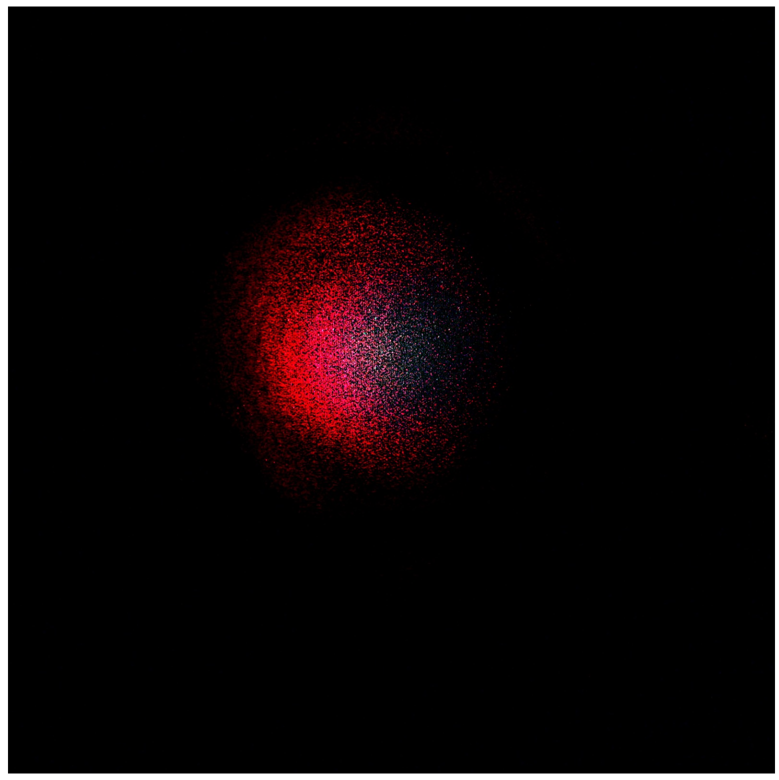
We tried multiple different attempts at this:

- Taking pictures while the block is cooling from $80°C$ and also while heating up

NOTE: We did not measure the temperature when the pictures were taken since we were seeing whether an interference pattern would emerge or not.

However, we were not able to obtain an interference pattern of any sort:

PUT RESULTANT HERE



We expect this could be due to a few things:

1. It is very difficult (without using the slots in the table) to properly line up the two split beams back to the camera. This could be resulting in poor interference
2. The metal block is not expanding (and deforming) as we expect. For exmaple, if it was a completely uniform deformation across the face, then we would not see “fringes” since that relies on a gradient of deformation
3. Since we have one of the bay doors open in order to point the heat gun at the block, external light sources is interfering with the experiment (we did try to solve this by turning all of the lights off, but we note that there were still computer lights on)

## 6.9 Week 4 Thursday 20/10/2022

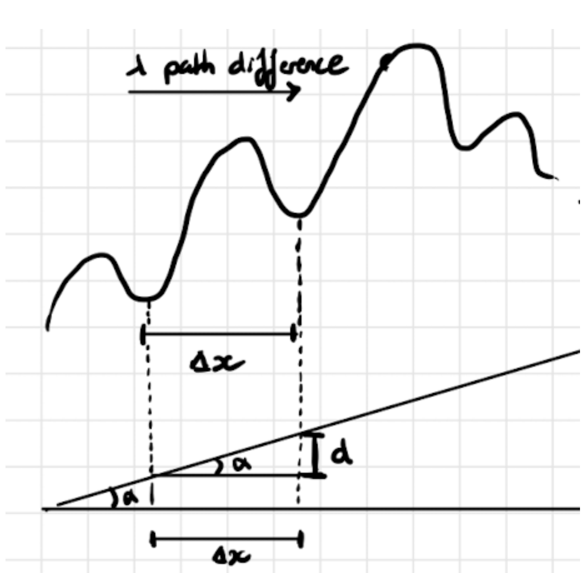
Looking through the derivation made in Week 3 Tuesday, I made a mistake in my derivation.

### 9:45-

The final equation relating the constant of proportionality of Voltage against $\alpha$, $k$, to the gradient of the Voltage against 1/fringe spacing plot, $m$, is actually:

$$k = \frac{2}{m\lambda}$$

i.e. $m$ is in the denominator rather than the numerator. To find the deformation required for a fringe to appear, i.e. the path difference of the outgoing light is $n\lambda$ and $n = 1$, which i call distance $d$:




$$d = \tan(\alpha)\Delta x \xrightarrow{\text{small angle approx}} d \approx \alpha\Delta x$$

- $y(x) = \frac{\lambda}{2}\frac{x}{\Delta x} = \tan\alpha \, x$
- $V = k\alpha$
- $\Delta x = \frac{\lambda}{2\tan\alpha} \simeq \frac{\lambda}{2\alpha} \quad (\alpha \ll 1)$
- $\frac{1}{\Delta x} = mV$

$\Delta x = \frac{\lambda}{2\alpha} \Rightarrow \alpha = \frac{\lambda}{2\Delta x} \quad \therefore V = \frac{k\lambda}{2\Delta x}$

$V$ → [m]

[m]

convert to m

$V \cdot \Delta x = \frac{1}{m}$ and $V\Delta x = \frac{k\lambda}{2}$

$V^{-1}m^{-1}$

$\Rightarrow m = \frac{2}{k\lambda} \quad \therefore k = \frac{2}{m\lambda}$ → voltage response of actuator

$d = \tan\alpha \, \Delta x \simeq \alpha \, \Delta x$

If we substitue in $\alpha = \frac{V}{k} = \frac{m\lambda V}{2}$ and $\Delta x = \frac{1}{mV}$, we get:

$$d = \frac{\lambda}{2}$$

This makes sense because the path difference of outgoing light is 2 times the defotmation of the surface, thus the path difference is $\lambda$, which is what we expect.

### 11:15-

We can now find the voltage response of the actuator. If we call the width of the screen $X$, we can find the maximum displacement, $D$, of the actuator. Using the first bullet point equation, we can substute these values in to get:

$$D = \frac{\lambda}{2}\frac{X}{\Delta x} = \frac{\lambda}{2}XmV,$$

We measured the width of the screen 3 times and took the weighted average, with a final value of $X = 75.66 \pm 0.005\text{mm}.$

NOTE: This is assuming that the screen is being pushed right at the edge (i.e. the entire white screen is being rotated from edge to edge)

Thus, for a voltage of $10V$ and using the gradient of the second run for rising voltage, $m = 25.2 \pm 0.5$, we get a maximum displacement of:

$$D = 6.03\mu m \xrightarrow{\text{range of}} 1 - 10\mu m$$

### Ball-park figure of how many fringes we expect to see:

$D = 6 \times 10^{-6}$ m, $\lambda \simeq 600$ nm

$d_{min} = \frac{\lambda}{2} \Rightarrow \frac{D}{d_{min}} = \frac{2D}{\lambda} \simeq 20 - 1 = 19$ fringes

Light only covers $\frac{1}{2} \sim \frac{1}{3}$ of the screen. Thus expected no. of seen fringes:

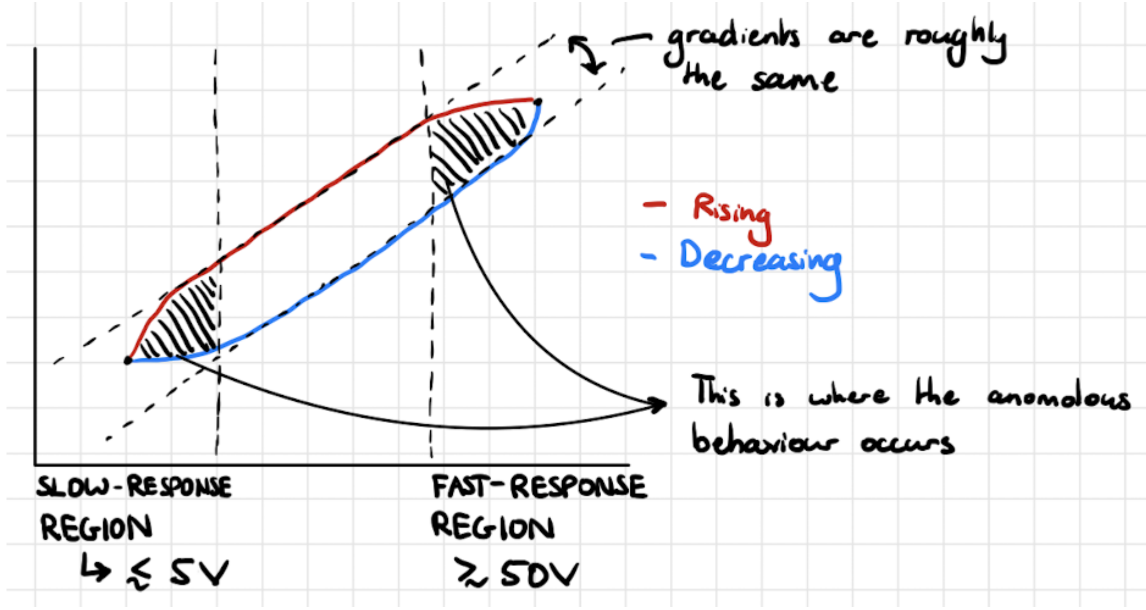
$19 \div 3 \simeq 6$ fringes

For 10V we see 4 fringes

This ballpark figure suggests that the range of max displacement of the actuator does indeed agree with the number of fringes we are seeing.

### 12:09-

### Why do we have a hysterisis and why does visibility get worse when decreasing voltage




The hysterisis arises due to different response regions of the PZT-actuator (this is consistent for all measurements).

- Slow-response region (3 possible reasons):
    1. The PZT actuator does not respond linearly at low voltages ($< 5V$). This could be due to how the crystal itself behaves at low voltages, or due to a response time to the change in voltage (would also explain why it does not respond linearly immediately when decreasing voltage)
    2. The technique itself is limiting the measurement of the fringe seperation at low voltages. Since the defomation of the plate at low volatges is comparable to that of the resolution of the technique (which is of order $\lambda$), and since we need discrete peaks in order to calculate the fringe seperation, the deformation does not pass a threshold required for accurate measurements to be made.
        a. We do not belive that this is the case since there is an obvious shift for ***all*** the data points, not just those for small deformations
    3. The voltmeter itself did not effectively change when at low voltages, i.e. it was not actually at 5V when set. This is much less likely as does not explain the why the curve is non-linear outside of the slow-response region.
- Fast-response region:
    1. When there is a large voltage (which appears to be around 55V) the dielectric constant of the PZT crystal becomes non-linear. This could be because at this voltage, the crystal no longer expands simply due to the potential difference across it, but an increase in temperature as well is causing it to expand more.

## 6.10 Week 5 Tuesday 25/10/2022

The conversion from metres to pixels is:

$$\mathrm{m} \to \mathrm{pix} = 465 \pm 11 \mathrm{pix\ m}^{-1}$$

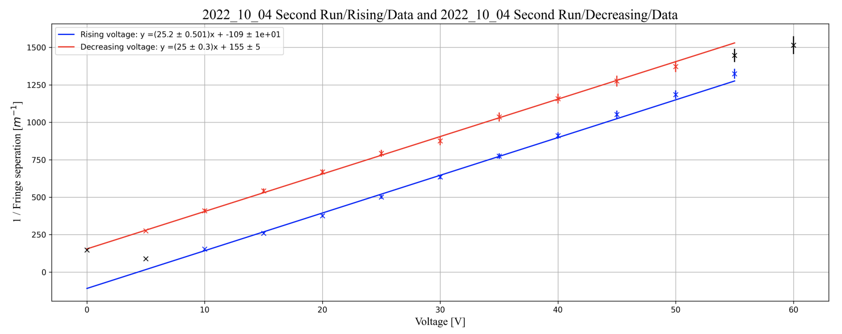
Thus we divide our values for the mean fringe spacing (which is in length of pixels) to get them in metres. We can then find the uncertainty of the final result by (note $\mathrm{f.s.}$ stands for fringe spacing):

$$\begin{cases} {}^{[\mathrm{pix}]}\mu_{\mathrm{f.s.}} &= \frac{\Delta x_1 + \Delta x_2 + \dots}{N} \\ {}^{[\mathrm{pix}]}\bar{\sigma}^2_{\mathrm{f.s.}} &= \frac{\sigma_{\mathrm{f.s.}}}{\sqrt{N}} \end{cases} \quad \text{and} \quad \begin{cases} \rho &= 465 \times 10^2 \mathrm{pix\ m}^{-1} \\ \sigma_\rho &= 11 \times 10^2 \mathrm{pix\ m}^{-1} \end{cases}$$

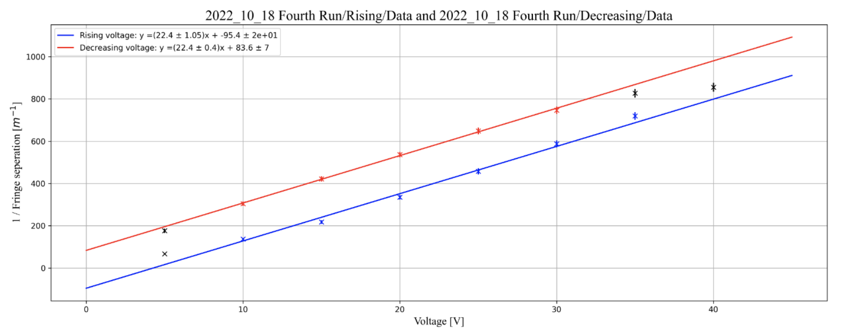
$$\begin{cases} {}^{[\mathrm{m}]}\mu_{\mathrm{f.s.}} &= \frac{{}^{[\mathrm{pix}]}\mu_{\mathrm{f.s.}}}{\rho} \\ {}^{[\mathrm{m}]}\sigma_{\mathrm{f.s.}} &= \sqrt{(\frac{1}{\rho})^2 \, {}^{[\mathrm{pix}]}\bar{\sigma}^2_{\mathrm{f.s.}} + (\frac{{}^{[\mathrm{pix}]}\mu_{\mathrm{f.s.}}}{\rho^2})^2 \sigma^2_\rho} \end{cases} \overset{1/\text{fringe spacing}}{\Longrightarrow} \begin{cases} y &= 1/{}^{[\mathrm{m}]}\mu_{\mathrm{f.s.}} \\ \sigma &= \left(\frac{1}{{}^{[\mathrm{m}]}\mu_{\mathrm{f.s.}}}\right)^2 {}^{[\mathrm{m}]}\sigma_{\mathrm{f.s.}} \end{cases}$$

```python
#average spacing in pixels
average_fringe_spacing = np.average(fringe_spacing)
uncertainty_fringe_spacing = np.std(fringe_spacing) / np.sqrt(len(fringe_spacing))

# pixels to metres
pix_to_m = 465e2
uncertainty_pix_to_m = 11e2

metre_fringe_spacing = average_fringe_spacing / pix_to_m
uncertainty_metre_fringe_spacing = np.sqrt((1/pix_to_m)**2*uncertainty_fringe_spacing**2 + (average_fringe_spacing/(pix_to_m**2))**2*uncertainty_pix_to_m**2)

axs.grid()
axs.set_xlim((np.min(data[:, 0]), np.max(data[:, 0])))

plt.tight_layout()
plt.savefig(save_folder + title, dpi=300, transparent=False)
plt.close()

return np.array((int(title), 1 / (metre_fringe_spacing),
                (1 / (metre_fringe_spacing) ** 2)
                * uncertainty_metre_fringe_spacing))
```

where $y$ is the data we plot on the graphs below. As a result of this propogation, the errors on each data point is much larger and is now roughly $\sim 10\%$. The resultant plots and lines of best fits are thus:




```
SAVGOL_FILTER_PARAMETERS_1 = {"00": 501, #savgol parameters for increasing voltage
                              "05": 151,
                              "10": 131,
                              "15": 101,
                              "20": 81,
                              "25": 71,
                              "30": 61,
                              "35": 51,
                              "40": 41,
                              "45": 41,
                              "50": 31,
                              "55": 21,
                              "60": 11,
                              "65": 11,
                              "70": 11,}
SAVGOL_FILTER_PARAMETERS_2 = {"00": 501, #savgol parameters for decreasing voltage
                              "05": 151,
                              "10": 131,
                              "15": 101,
                              "20": 81,
                              "25": 71,
                              "30": 61,
                              "35": 51,
                              "40": 41,
                              "45": 41,
                              "50": 31,
                              "55": 21,
                              "60": 11,
                              "65": 11,
                              "70": 11,}
PEAK_PROMINENCE = {
    "Rising": 0.05,
    "Decreasing": 0.02
}
```




```
SAVGOL_FILTER_PARAMETERS_1 = {"00": 801, #savgol parameters for increasing voltage
                              "05": 201,
                              "10": 131,
                              "15": 101,
                              "20": 81,
                              "25": 71,
                              "30": 61,
                              "35": 51,
                              "40": 41,
                              "45": 41,
                              "50": 31,
                              "55": 21,
                              "60": 11,
                              "65": 11,
                              "70": 11,}
SAVGOL_FILTER_PARAMETERS_2 = {"00": 701, #savgol parameters for decreasing voltage
                              "05": 121,
                              "10": 111,
                              "15": 95,
                              "20": 71,
                              "25": 61,
                              "30": 51,
                              "35": 51,
                              "40": 41,
                              "45": 41,
                              "50": 31,
                              "55": 21,
                              "60": 11,
                              "65": 11,
                              "70": 11,}
PEAK_PROMINENCE = {
    "Rising": 1,
    "Decreasing": 1
}
```

The reduced chi-square of the points is calculated as:

$$\chi^2_{\text{reduced}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2$$

```python
def reduced_chi_square(data, param):
    chi_square_total = 0
    for datum in data:
        chi_square_total += (((linear_function(datum[0], *param) -
                                datum[1]) / datum[2]) ** 2)

    return chi_square_total / len(data)
```

Which gives:

- Second Run:
  - Rising reduced chi squared: 1.99

    - Decreasing reduced chi squared:  0.744
    - Rising voltage: $y = (25.3 \pm 0.48)x + (-112 \pm 8.7)$
    - Decreasing voltage: $y = (25.4 \pm 0.37)x + (149 \pm 6.5)$
    - Rising voltage: $y = (5.95e - 07 \pm 1.1e - 08)x + (-2.64e - 06 \pm 2e - 07)$
    - Decreasing voltage: $y = (5.96e - 07 \pm 8.6e - 09)x + (3.51e - 06 \pm 1.5e - 07)$
- Fourth Run:
    - Rising reduced chi squared:  2.38
    - Decreasing reduced chi squared:  0.131
    - Rising voltage: $y = (22.8 \pm 0.75)x + (-112 \pm 11)$
    - Decreasing voltage: $y = (22.6 \pm 0.35)x + (80.1 \pm 6)$
    - Rising voltage: $y = (5.36e - 07 \pm 1.8e - 08)x + (-2.62e - 06 \pm 2.6e - 07)$
    - Decreasing voltage: $y = (5.31e - 07 \pm 8.3e - 09)x + (1.88e - 06 \pm 1.4e - 07)$

The chi-sqaured for the second run are much better than the fourth run. All gradients are similar to each other and the y-intercepts are basically the negative of each other for increasing and decreasing voltage.